Motion Guided 3D Pose Estimation from Videos

Jingbo Wang[1[0000−0001−9700−6262]], Sijie Yan[1[0000−0003−4398−0590]], Yuanjun Xiong[2[0000−0002−6391−4921]], and Dahua Lin[1[0000−0002−8865−7896]]

[1] The Chinese University of Hong Kong
{jbwang,ys016,dhlin}@ie.cuhk.edu.hk
[2] AWS/Amazon AI
yuanjx@amazon.com

Abstract. We propose a new loss function, called motion loss, for supervising models for monocular 3D Human pose estimation from videos. It works by comparing the motion pattern of the prediction against ground truth key point trajectories. In computing motion loss, we introduce pairwise motion encoding, a simple yet effective representation for keypoint motion. We design a new graph convolutional network architecture, U-shaped GCN (UGCN). It captures both short-term and long-term motion information to fully leverage the supervision from the motion loss[1]. We experiment training UGCN with the motion loss on two large scale benchmarks: Human3.6M and MPI-INF-3DHP. Our models surpass other state-of-the-art models by a large margin. It also demonstrates strong capacity in producing smooth 3D sequences and recovering keypoint motion.

Keywords: 3D Pose Estimation, Motion Loss, Graph Convolution

1 Introduction

3D human pose estimation aims at reconstructing 3D body keypoints from their 2D projections, such as images [36,14,33,26], videos [4,35], 2D pose [17,27,15], or their combination [24,34]. Unlike the 2D pose estimation, this problem is ill-posed in the sense that the lack of depth information in the 2D projections input leads to ambiguities. To obtain the perception of depth, recent works [11,28] utilized multiple synchronized cameras for observing objects from different angles and have achieved considerable progress. However, compared with monocular methods, multi-view methods are not practical in reality because of their strict prerequisites for devices and environments.

In recent years, video-based 3D human pose estimation [2,15,16,5] receives attention quickly. Taking a video as input, models are able to perceive the 3D structure of an object in motion and better infer the depth information for 3D pose estimation in each frame. Unlike image-based models, video-based models [15,2] are supervised by a long sequence of 3D pose, which increase the dimensionality

[1] Codes and models at **http://wangjingbo.top/papers/ECCV2020-Video-Pose/MotionLossPage.html**.

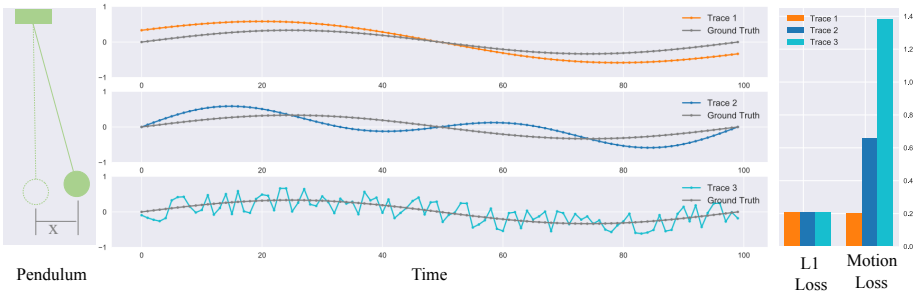

Fig. 1. A toy sample, the location estimation of pendulum motion. We show the horizontal location as time varies, a sine curve, denoted in gray, and three estimated traces, denoted in blue, orange and cyan. They have the same ℓ_1 mean distance to the groundtruth but have different temporal structure. Which estimated trace better describes the pendulum motion? The loss under different matrices is also shown in the right figure. Obviously, motion loss is good at answering the above question.

of solution space by hundreds of times. In most existing works, the common loss function for supervising 3D pose estimation models is *Minkowski Distance*, such as ℓ_1-loss and ℓ_2-loss. It independently computes the overall location error of the predicted keypoints in 3D space with respect to their ground-truth locations.

However, there is a critical limitation for the Minkowski Distance. It does not consider the similarity of temporal structure between the estimated pose sequence and the ground-truth. We illustrate this issue by a toy sample, the trace estimation of a pendulum motion. It is similar to pose estimation, but only includes one "joint". We compare three estimated trajectories of pendulum motion in Figure.1. The first trace function has a shape similar to the ground-truth. The second one has a different tendency but still keep smoothness. And the last curve just randomly fluctuates around the ground-truth. Both of them have the same ℓ_1 mean distance to the ground-truth but have various temporal structures. Because the Minkowski Distance is calculated independently for each moment, it failed to examine the inner dependencies of a trajectory.

The keypoints in a pose sequence describe the human movement, which are strongly correlated especially in the time. Under the supervision of Minkowski Distance as the loss, same as the above toy sample, it is difficult for models to learn from the motion information in the ground-truth keypoint trajectories and thus hard to obtain natural keypoints movement in the model's prediction due to the high dimensional solution space.

We address this issue by proposing *motion loss*, a novel loss function that explicitly involves motion modeling into the learning. Motion loss works by requiring the model to reconstruct the keypoint motion trajectories in addition to the task of reconstructing 3D locations of keypoints. It evaluates the motion reconstruction quality by computing the difference between predicted keypoint locations and the ground-truth locations in the space of a specific representation

called *motion encoding*. The motion encoding is built as a differentiable operator in the following manner. We first roughly decompose a trajectory into a set of pairwise coordinate vectors with various time intervals corresponding to different time scales. A basic differentiable binary vector operator, for instance, subtraction, inner product or cross product, is applied to each pair. Then the obtained results are concatenated to construct the full motion encoding. Though simple, this representation is shown in the Figure 1 (taking subtraction operator for example) to be effective in assessing the quality of the temporal structure. The difference in motion loss values clearly distinguishes the motion reconstruction quality of the three trajectories. By applying it to the training of 3D pose estimation models, we also observe that motion loss can significantly improve the accuracy of 3D pose estimation.

To estimate the pose trajectories with reasonable human movements, the 3D pose estimation model must have the capacity to model motion in both short temporal intervals and long temporal ranges, as human actions usually have varying speeds over time. To achieve this property we propose a novel graph convolutional network based architecture for 3D pose estimation model. We start by repurposing an ST-GCN [38] model, initially proposed for skeleton-based action recognition, to take as input 2D pose sequences and output 3D pose sequences. Inspired by the success of U-shaped CNNs used in semantic segmentation and object detection, we construct a similar U-shaped structure on the temporal axis of the ST-GCN [38] model. The result is a new architecture, called *U-shaped GCN* (UGCN), with strong capacity in capturing both short-term and long-term temporal dependencies, which is essential in characterizing the keypoint motion.

We experiment the motion loss and UGCN for video-based 3D pose estimation from 2D pose on two large scale 3D human pose estimation benchmarks: Human3.6M [9] and MPI-INF-3DHP [18]. We first observe a significant boost in position accuracy when the motion loss is used in training. This corroborates the importance of motion-based supervision. When the motion loss is combined with UGCN, our model surpasses the current state of the art models in terms of location accuracy by a large margin. Besides improved location accuracy, we also observe that UGCN trained with the motion loss is able to produce smooth 3D sequences without imposing any smoothness constraint during training or inference. Our model also halves the velocity error [27] compared with other state of the art models, which again validates the importance of having motion information in the supervision. We provide detailed ablation study and visualization to further demonstrate the potential of our model.

2 Related work

3D pose estimation. Before the era of deep learning, early methods for 3D human pose estimation were based on handcraft features [29,9,8]. In recent years, most works depend on powerful deep neural networks and achieve promising improvements, which can be divided into two types.

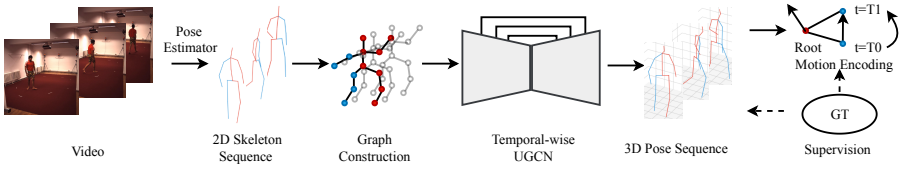

Fig. 2. Overview of our proposed pipeline for estimating 3D poses from consecutive 2D poses. We structure 2D skeletons by a spatial-temporal graph and predict 3D locations via our U-shaped Graph Convolution Networks (UGCN). The model is supervised in the space of motion encoding.

In the first type, estimators predict 3D poses from 2D image directly [35,26,33]. For example, [14] jointly regresses joint locations and detects body parts by sliding window on the image. [35] directly regresses the 3D pose from an aligned spatial-temporal feature map. [26] predicts per voxel likelihoods for each joint based on the stacked hourglass architecture. [33] utilizes an auto-encoder to learn a latent pose representation for modeling the joint dependencies.

Another typical solution builds on a two-stage pipeline [17,27,2,15]. Thereon, a 2D pose sequence is firstly predicted by a 2D pose estimator from a video frame by frame and lifted to 3D space by another estimator. For instance, [17] proposes a simple baseline composed of several fully-connected layers, which takes a single 2D pose as input. [27] generates 3D poses from 2D keypoint sequences by a temporal-convolution method. [2] introduces a local-to-global network based on graph convolution layers. [15] factorizes a 3D pose sequence into trajectory bases and train a deep network to regress the trajectory coefficient matrix.

Although the appearance information is dropped in the first stage, the data dimension is dramatically decreased as well, which makes long-term video-based 3D pose estimation possible. Our method also builds on the two-stage pipeline.

Graph convolution. Modeling skeleton sequence via spatial-temporal graphs(st-graph) [38] and performing graph convolution thereon has significantly boosted the performance in many human understanding tasks including action recognition [38], pose tracking [23] and motion synthesis [37]. The designs for graph convolution mainly fall into two stream: spectral based [6,12] and spatial based [1,22]. They extended standard convolution to irregular graph domain by Fourier transformation and neighborhood partitioning respectively. Following [38], we perform spatial graph convolution on skeleton sequences represented by st-graphs.

3 Approach

Figure. 2 illustrates our pipeline for estimating 3D pose sequences. Given the 2D projections of a pose sequence estimated from a video $P = \{\boldsymbol{p}_{t,j} | t = 1, ..., T; j = 1, ..., M\}$, we aim to reconstruct their 3D coordinates $S = \{\boldsymbol{s}_{t,j} | t = 1, ..., T; j = 1, ..., M\}$, where T is the number of video frames, M is the number of human

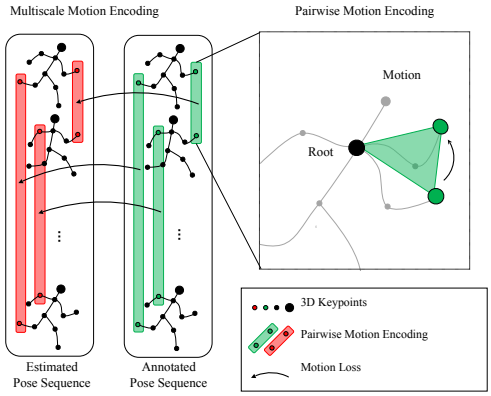

Fig. 3. Motion loss. By concatenating pairwise cross-product vectors between the coordinate vectors of the same joints across time with various intervals, we construct multi-scale motion encoding on pose sequences. The motion loss requires the model to reconstruct this encoding. It explicitly involves motion modeling into learning.

joints, $\boldsymbol{p}_{t,j}$ and $\boldsymbol{s}_{t,j}$ are vectors respectively representing the 2D and 3D locations of joint j in the frame t. We structure these 2D keypoints by a spatial-temporal graph and predict their 3D locations via our U-shaped Graph Convolution Networks (UGCN). The model is supervised by a multiscale motion loss and trained in an end-to-end manner.

3.1 Motion Loss

In this work, motion loss is defined as the distance in the space of motion. Therefore, a motion encoder is required for projecting skeleton sequences to this space. Though there are myriad possible designs, we empirically sum up a few guiding principles: differentiability, non-independence, and multi-scale. Differentiability is the prerequisite for the end-to-end training. And the calculation should be across time for modeling the temporal dependencies, *i.e.*, non-independence. Since the speed of motion is different, multi-scale modeling is also significant. In this section, we introduce how we design a simple but effective encoding, named *pairwise motion encoding*.

Pairwise motion encoding. We first consider the simplest case: the length of the pose sequences is 2. The motion encoding on the joint j can be denoted as:

$$\boldsymbol{m}_j = \boldsymbol{s}_{0,j} \star \boldsymbol{s}_{1,j}, \tag{1}$$

where $\star$ can be any differentiable binary vector operator, such as subtraction, inner-product and cross-product. In the common case, the pose sequence is

longer. We can expand an extra dimension in the motion encoding:

$$\boldsymbol{m}_{t,j} = \boldsymbol{s}_{t,j} \star \boldsymbol{s}_{t+1,j}. \tag{2}$$

Note that, this representation only models the relationship between two adjacent moments. Since the speed of human motion has a large variation range, it inspires us to encode human motion on multiple temporal scales:

$$\boldsymbol{m}_{t,j,\tau} = \boldsymbol{s}_{t,j} \star \boldsymbol{s}_{(t+\tau),j}. \tag{3}$$

where τ is the time interval. As shwon in Figure. 3, to caculate the motion loss of the full pose sequence, we compute the ℓ_1 Distance on the encoded space for all joints, moments and several time intervals. Mathematically, we have:

$$L_m = \frac{1}{\mathbb{T}} \sum_{\tau \in \mathbb{T}} \sum_{t=1}^{T-\tau} \sum_{j=1}^{M} \left\| \boldsymbol{m}_{t,j,\tau} - \boldsymbol{m}^{gt}_{t,j,\tau} \right\|, \tag{4}$$

where the interval set $\mathbb{T}$ includes different τ for multiple time scales. Pairwise motion encoding decomposes a trajectory into coordinate pairs and extracts features for each pair by a differentiable operation $\star$. As the first work to explore the supervision of motion for 3D pose estimation, intuitively, we choose the three most basic operations in the experiments: **subtraction, inner-product, and cross-product**. And we conducted extensive experiments to evaluate the effectiveness of these encoding methods in Section 4.3.

Loss Function. The motion loss only considers the second-order correlations in the formulation of pairwise motion encoding, while the absolute location information is absent. Therefore, we add a traditional reconstruction loss term to the overall training objectives:

$$L_p = \sum_{t=1}^{T} \sum_{j=1}^{M} \left\| \boldsymbol{s}_{t,j} - \boldsymbol{s}^{gt}_{t,j} \right\|_2^2. \tag{5}$$

The model is supervised in an end-to-end manner with the combined loss:

$$L = L_p + \lambda L_m, \tag{6}$$

where λ is a hyper parameter for balancing two objectives.

3.2 U-shaped Graph Convolutional Networks

Intuitively, the 3D pose estimator needs stronger long-term perception for exploring the motion priors. Besides that, keeping the spatial resolution is also required by estimating 3D pose accurately. Therefore, we represent the skeleton sequence as a *spatial temporal graph* [38] to maintain their topologies, and aggregating information by an *U-shaped graph convolution network (UGCN)*.

Graph Modeling It is an ill-posed problem to recover the 3D location of a keypoint from its 2D coordinates independently. In general, the information from other keypoints, especially the neighboring ones, play essential roles in 3D pose reconstruction. To model the relationship with these relative keypoints, it is natural to organize a skeleton sequence via a *spatial temporal graph* (st-graph) [38]. In particular, a st-graph G is determined by a node set and an edge set. The node set $V = \{v_{t,j} | t = 1, \ldots, T, j = 1, \ldots, M\}$ includes all the keypoints in a sequence of pose. And the edge set E is composed of two parts: one for connecting adjacent frames on each joint, one for the connecting endpoint of each bone in every single frame. These edges construct the temporal dependencies and spatial configuration together. Then, a series of graph convolution operations are conducted on this graph.

Graph Convolution. In this work, we adopt *spatial temporal graph convolution (st-gcn)* [38] as the basic unit to aggregate features of nodes on a st-graph. It can be regarded as a combination of two basic operations: a spatial graph convolution and a temporal convolution. The temporal convolution $Conv_t$ is a standard convolution operation applied on the temporal dimension for each joint, while the spatial graph convolution $Conv_g$ is performed on the skeleton for each time position independently. Given an input feature map f_{in}, the output of two operations can be written as:

$$f_s = Conv_g(f_{in}) \tag{7}$$

$$f_{out} = Conv_t(f_s) \tag{8}$$

, where f_s is the output of the spatial graph convolution. We follow the formulation of spatial graph convolution in [38]. And more details are in our supplementary materials.

Network structure. As shown in Figure 4, the basic units for building networks are st-gcn blocks, which include five basic operations: a spatial graph convolution, a temporal convolution, a batch normalization, a dropout and an activation function ReLU. Our networks are composed of three stages: downsampling, upsampling, and merging.

In the downsampling stage, we utilize 9 st-gcn blocks for aggregating temporal features. In addition, we set $stride = 2$ for the second, fourth, sixth, and eighth st-gcn blocks to increase the receptive field in the time dimension. This stage embeds the global information of the full skeleton sequence.

The upsampling stage contains four st-gcn blocks. Each block is followed by an upsampling layer. Thanks to the regular temporal structure in st-graph, the upsampling in the time dimension can be simply implemented with the following formula:

$$f_{up}(v_{t,j}) = f_{in}(v_{t',i}), \tag{9}$$

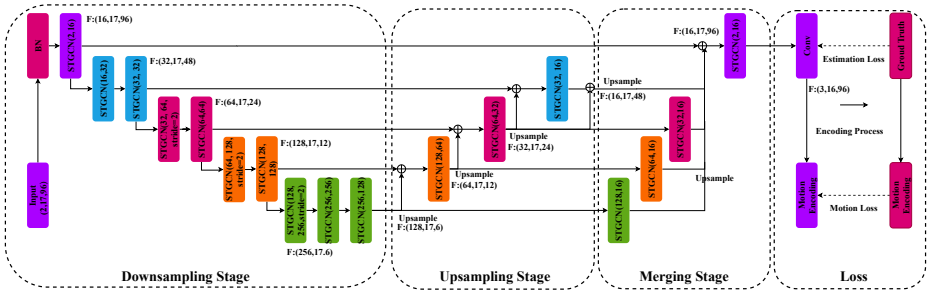

Fig. 4. Network structure. We proposed a U-shaped graph convolution network (UGCN) as the backbone of our pose estimation model to incorporate both local and global information with a high resolution. This network consists of three stages: downsampling, upsampling and merging. The network first aggregates long-range information by temporal pooling operations in the downsampling stage. And then recovers the resolution by upsampling layers. To keep the low-level information, the features in the downsampling stage are also added to the upsample branch by some shortcuts. Finally, the multi-scale feature maps are merged to predicted 3D skeletal joints. In this way, UGCN incorporates both short-term and long-term information, making it an ideal fit for the supervision of the motion loss.

where $t' = \lfloor \frac{t}{2} \rfloor$. With successive upsampling operations, the temporal resolution gradually recovers and the global information spread to the full graph. Since the 2D inputs are projections of 3D outputs, the low-level information may provide strong geometric constraints for estimating 3D pose. It motivated us to keep low-level information in the networks. Thus, we add features in the first stage to the upsampling stage with the same temporal resolution.

In the merging stage, the feature maps with various time scales in the second stage are transformed to the same shape and fused to obtain the final embedding. Obviously, this embedding contains abundant information on multiple temporal scales.

In the end, the 3D coordinate for each keypoint is estimated by a st-gcn regressor. This model is supervised by the motion loss in an end-to-end manner. Other details have been depicted in the Figure 4.

Training & inference. We use st-gcn blocks with the temporal kernel size of 5 and the dropout rate of 0.5 as our basic cells to construct a UGCN. The networks take as input a 2D pose sequence with 96 frames. We perform horizontal flip augmentation at the time of training and testing. Considering the various value ranges of different motion encoding in Section 3.1, we normalize the inner-product and cross product encoding by the temporal-wise mean value before computing motion loss. Based on this normalization, we can set $\lambda = 1$ to balance the reconstruction loss and our motion loss conveniently. We optimize the model using Adam for 110 epochs with the batch size of 256 and the initial learning rate of 10^{-2}. We decay the learning rate by 0.1 after 80, 90 and 100 epochs. To

avoid the overfitting, we set the weight decay factor to 10^{-5} for parameters of convolution layers.

In the inference stage, we apply the sliding window algorithm with the step length of 5 to estimate a variable-length pose sequence with fixed input length, and average all results on different time positions.

4 Experiments

We evaluate models on two large-scale datasets for 3D pose estimation: Human3.6M and MPI-INF-3DHP. In particular, we first perform detailed ablation studies on the Human3.6M dataset to examine the effectiveness of the proposed components. To exclude the interference of 2D pose estimator, all experiments in this ablation study take 2D ground truth as input. Then, we compare the estimated results of UGCN with other state-of-the-art methods on two datasets. All experiments are conducted on PyTorch tools with one single TITANX GPU.

4.1 Dataset

Human3.6M: Human3.6M [10] is a large-scale indoor dataset for 3D human pose estimation. This widely used dataset consists of 3.6 million images which are captured from 4 different cameras. There are 11 different subjects and 15 different actions in this dataset, such as "Sitting", "Walking", and"Phoning". The 3D ground truth and all parameters of the calibrated camera systems are provided in this dataset. However, we do not exploit the camera parameters in the proposed approach. Following the recent works, we utilize (S1, S5, S6, S7, S8) for training and (S9, S11) for testing. The video from all views and all actions are trained by a single model. For this dataset, we conduct ablation studies based on the ground truth of 2D skeleton. Besides that, we also report the results of our approach taking as input predicted 2D poses. from widely used pose estimators.

Table 1. Performance of our UGCN model supervised by motion loss with different basic operators and time intervals. The empty set $\varnothing$ denotes that the motion loss is not utilized. The best MPJPE is achieved by the cross product operator with interval of 12.

Interval set $\mathbb{T}$	$\varnothing$	{2}	{4}	{8}	{12}	{16}	{24}	{36}	{48}
Subtraction	32.0	31.4	30.8	29.7	**28.9**	29.3	30.6	31.8	32.8
Inner Product	32.0	31.8	31.7	31.0	30.2	**29.8**	31.2	32.6	33.7
Cross Product	32.0	31.2	30.4	28.2	**27.1**	28.3	30.2	31.6	32.7

MPI-INF-3DHP: MPI-INF-3DHP [19] is a recently released 3D human pose estimation dataset. And this dataset is captured in both indoor environment and in-the-wild outdoor environment. Similar to Human3.6M, this dataset also provides videos from different cameras, subjects, and actions.

Table 2. We select the 4 best time intervals according to the Table.1, and add them to the interval set one by one. More keypoint pairs with different intervals involve the calculation of motion encoding. The MPJPE is improved in this process.

Operator	$\tau = 8$	$\tau = 12$	$\tau = 16$	$\tau = 24$	# Time Scales	MPJPE(mm)
Cross Product		✓			1	27.1
Cross Product	✓	✓			2	26.3
Cross Product	✓	✓	✓		3	25.7
Cross Product	✓	✓	✓	✓	4	**25.6**

4.2 Evaluation Metric

For both Human3.6M and MPI-INF-3DHP dataset, we report the *mean per joint position error(MPJPE)* [15,27,2] as the evaluation metric. In general, there are two protocols, *Protocol-1* and *Protocol-2*, used in the previous works to evaluate 3D pose estimation. Metric Protocol-1 first aligns the root joint(central hip) and then calculates the average Euclidean distance of the estimated joints. While in the Protocol-2, the estimated results are further aligned to the ground truth via a rigid transformation before computing distance.

In MPI-INF-3DHP, we evaluate models under two additional metrics. The first one is the area under the curve (AUC) [40] on the percentage of correct keypoints(PCK) score for different error thresholds. Besides, PCK with the threshold of 150mm is also reported.

4.3 Ablation Study

In this section, we demonstrate the effectiveness of the proposed UGCN and our motion loss on the Human3.6M dataset. Experiments in this section directly take 2D ground-truth as input to eliminate the interference of 2D pose estimator.

Effect of motion loss. We start our ablation study from observing the impact of the temporal interval τ in the single scale motion loss. In other words, the interval set for motion loss has only one element. The value of this element controls the temporal scale of motion loss. We conduct experiments on three binary operators proposed in Section 3.1, *i.e.* subtraction, inner-product and cross-product.

As shown in Table 1, the cross-product achieves the lowest MPJPE error with almost all temporal intervals. Besides, the MPJPE error decrease first and then increase, and reduce the error by 4.9mm (from 32.0 to 27.1) with the time interval of 12 and the cross-product encoding. There are two observations. First, compared to the result without motion term (denoted as $\varnothing$), even the temporal interval is large (24 frames), the performance gain is still positives. It implies that motion prior is not momentary. And the model might need long-term perception for better capturing the motion information. Second, motion loss boosts the performance with temporal interval τ in a large variation range (2$\sim$36 frames), which means the time scale of motion priors is also various.

Thus, it is reasonable to adopt motion loss with multiple time intervals. We select four best τ as candidates and adopt the most effective binary operator in Table. 1, cross-product. The experimental results have been depicted in Table. 2. Under the supervision of multiscale motion loss, our model decrease the MPJPE by $1.5mm$ ($27.1 \rightarrow 25.6$).

Table 3. We remove all downsampling and upsampling operations from the standard UGCN, and add them back pair by pair. The MPJPE performance of our system increases remarkably in this process. With motion loss, the achieved gain is even large.

# Downsample & Upsample	0	1	2	4	Δ
UGCN w/o Motion Loss	38.6	37.2	36.9	32.0	6.6
UGCN + Motion Loss ($\mathbb{T} = \{12\}$)	36.9	34.8	33.7	**27.1**	**9.8**
Δ	1.7	2.4	3.2	4.9	-

Table 4. We explore the importance of each individual component by removing them from standard setting. The increased MPJPE error for each module is listed below.

Backbone	MPJPE(mm)	Δ
UGCN	32.0	-
UGCN w/o Spatial Graph	39.2	7.2
UGCN w/o Merging Stage	32.5	0.5
UGCN + Motion Loss	25.6	-
UGCN + Motion Loss w/o Merging Stage	28.4	2.8

Table 5. The MPJPE performance of our system with different supervision. Combining motion loss functions with different basic operators does not bring obvious improvement.

Loss Function	Interval set $\mathbb{T}$	MPJPE(mm)	Δ
-	$\varnothing$	32.0	-
Derivative loss [30]	{1}	31.6	0.4
Cross product	{12}	27.1	4.9
Subtraction+ Cross product	{12}	27.1	4.9
Subtraction + Inner + Cross product	{12}	27.1	4.9

Design choices in UGCN. We first examine the impact of the U-shaped architecture. We remove all downsampling and upsampling operations from the standard UGCN, and add them back pair by pair. The experimental results have been depicted in Table. 3. It can be seen that U-shaped structure brings

significant improvement ($6.6mm$) to UGCN. This structure even leads to a larger performance gain ($9.8mm$) with the supervision of motion loss. And the gap caused by motion loss is growing with the increasing number of downsampling and upsampling. These results validate our assumption: the motion loss requires long-term perception.

We also explore other design choices in the UGCN. As shown in Table. 4, the spatial configuration bring $7.2mm$ improvement. Removing the merging stage only slightly enlarge the error. However, when the model is supervised by motion loss, the performance drop is more remarkable ($0.5mm$ vs. $2.8mm$). That is to say, multiscale temporal information is important to the learning of motion prior.

Design choices in motion loss. The formula of offset encoding is similar to the Derivative Loss [30] which regularizes the joint offset between adjacent frames. This loss is under the the hypothesis that the motion is smooth between the neighborhood frames. We extend it to our motion loss formulation. Since only short-term relation is considered, the improvement achieved by Derivative Loss is minor. Then we compare the results of our method supervised by the motion loss with different combination of the proposed binary operators. The results have been shown in Table. 5. The combination of these three representations is not able to bring any improvement. Therefore, we adopt cross-product as the pairwise motion encoder in the following experiments.

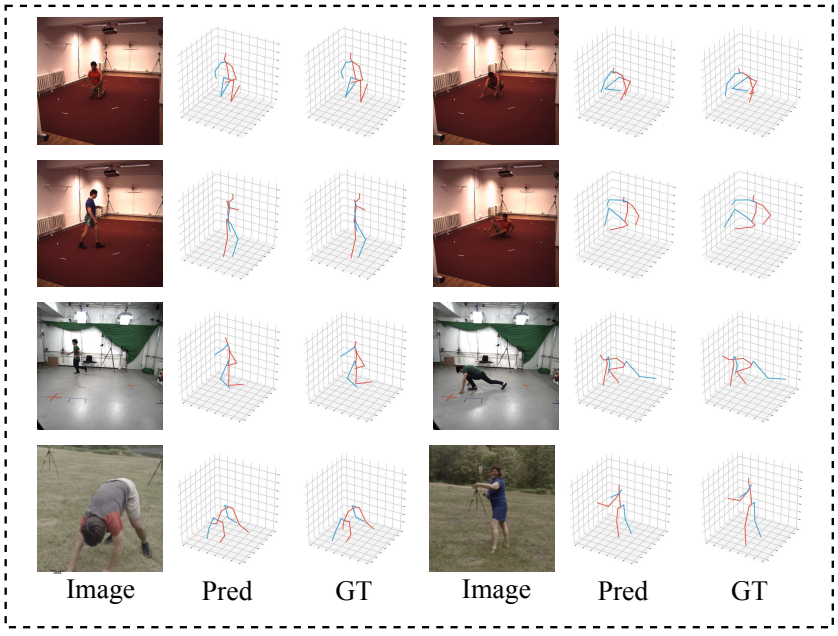

Fig. 5. Visulation results of our full system on Human3.6M and MPI-INF-3DHP.

Table 6. Results showing the errors action-wise on Human3.6M under Protocol-1 and Protocol-2. (CPN) and (HRNET) respectively indicates the model trained on 2D poses estimated by CPN [3], and HR-Net [31]. † means the methods adopt the same refine module as [2].

Protocol 1	Dir.	Disc.	Eat.	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Somke	Wait	WalkD.	Walk	WalkT.	Ave.
Zhou [39]	54.8	60.7	58.2	71.4	62.0	65.5	53.8	55.6	75.2	111.6	64.1	66.0	51.4	63.2	55.3	64.9
Martinez [17]	51.8	56.2	58.1	59.0	69.5	78.4	55.2	58.1	74.0	94.6	62.3	59.1	65.1	49.5	52.4	62.9
Sun [32]	52.8	54.8	54.2	54.3	61.8	67.2	53.1	53.6	71.7	86.7	61.5	53.4	61.6	47.1	53.4	59.1
Fang [7]	50.1	54.3	57.0	57.1	66.6	73.3	53.4	55.7	72.8	88.6	60.3	57.7	62.7	47.5	50.6	60.4
Pavlakos [25]	48.5	54.4	54.4	52.0	59.4	65.3	49.9	52.9	65.8	71.1	56.6	52.9	60.9	44.7	47.8	56.2
Lee [13]	43.8	51.7	48.8	53.1	52.2	74.9	52.7	44.6	56.9	74.3	56.7	66.4	68.4	47.5	45.6	55.8
Hossain [30]	48.4	50.7	57.2	55.2	63.1	72.6	53.0	51.7	66.1	80.9	59.0	57.3	62.4	46.6	49.6	58.3
Lee [13](F=3)	40.2	49.2	47.8	52.6	50.1	75.0	50.2	43.0	55.8	73.9	54.1	55.6	58.2	43.3	43.3	52.8
Dabral [5]	44.8	50.4	44.7	49.0	52.9	61.4	43.5	45.5	63.1	87.3	51.7	48.5	52.2	37.6	41.9	52.1
Pavllo [27]	45.2	46.7	43.3	45.6	48.1	55.1	44.6	44.3	57.3	65.8	47.1	44.0	49.0	32.8	33.9	46.8
Cai [2]†	44.6	47.4	45.6	48.8	50.8	59.0	47.2	43.9	57.9	61.9	49.7	46.6	51.3	37.1	39.4	48.8
Lin [15]	42.5	44.8	42.6	44.2	48.5	57.1	42.6	41.4	56.5	64.5	47.4	43.0	48.1	33.0	35.1	46.6
UGCN(CPN)	41.3	43.9	44.0	42.2	48.0	57.1	42.2	43.2	57.3	61.3	47.0	43.5	47.0	32.6	31.8	45.6
UGCN(CPN)†	40.2	42.5	42.6	41.1	46.7	56.7	41.4	42.3	56.2	60.4	46.3	42.2	46.2	31.7	31.0	44.5
UGCN(HR-Net)	**38.2**	**41.0**	**45.9**	**39.7**	**41.4**	**51.4**	**41.6**	**41.4**	**52.0**	**57.4**	**41.8**	**44.4**	**41.6**	**33.1**	**30.0**	**42.6**

Protocol 2	Dir.	Disc.	Eat.	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Somke	Wait	WalkD.	Walk	WalkT.	Ave.
Martinez [17]	39.5	43.2	46.4	47.0	51.0	56.0	41.4	40.6	56.5	69.4	49.2	45.0	49.5	38.0	43.1	47.7
Sun [32]	42.1	44.3	45.0	45.4	51.5	53.0	43.2	41.3	59.3	73.3	51.0	44.0	48.0	38.3	44.8	48.3
Fang [7]	38.2	41.7	43.7	44.9	48.5	55.3	40.2	38.2	54.5	64.4	47.2	44.3	47.3	36.7	41.7	45.7
Lee [13]	38.0	39.3	46.3	44.4	49.0	55.1	40.2	41.1	53.2	68.9	51.0	39.1	56.4	33.9	38.5	46.2
Pavlakos [25]	34.7	39.8	41.8	38.6	42.5	47.5	38.0	36.6	50.7	56.8	42.6	39.6	43.9	32.1	36.5	41.8
Hossain [30]	35.7	39.3	44.6	43.0	47.2	54.0	38.3	37.5	51.6	61.3	46.5	41.4	47.3	34.2	39.4	44.1
Pavllo [27]	34.1	36.1	34.4	37.2	36.4	42.2	34.4	33.6	45.0	52.5	37.4	33.8	37.8	25.6	27.3	36.5
Dabral [5]	**28.0**	**30.7**	39.1	34.4	37.1	44.8	**28.9**	**31.2**	**39.3**	60.6	39.3	**31.1**	37.8	25.3	28.4	36.3
Cai [2]†	35.7	37.8	36.9	40.7	39.6	45.2	37.4	34.5	46.9	50.1	40.5	36.1	41.0	29.6	33.2	39.0
Lin [15]	32.5	35.3	34.3	36.2	37.8	43.0	33.0	32.2	45.7	51.8	38.4	32.8	37.5	25.8	28.9	36.8
UGCN(CPN)	32.9	35.2	35.6	34.4	36.4	42.7	31.2	32.5	45.6	50.2	37.3	32.8	36.3	26.0	23.9	35.5
UGCN(CPN)†	31.8	34.3	35.4	33.5	35.4	41.7	31.1	31.6	44.4	49.0	36.4	32.2	35.0	**24.9**	23.0	34.5
UGCN(HR-Net)	28.4	32.5	**34.4**	**32.3**	**32.5**	**40.9**	30.4	29.3	42.6	**45.2**	**33.0**	32.0	**33.2**	**24.2**	**22.9**	**32.7**

Table 7. Results show the velocity error of our methods and other state-of-the-arts on Human3.6M. Our result without motion loss is denoted as (*).

MPJVE	Dir.	Disc.	Eat.	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Somke	Wait	WalkD.	Walk	WalkT.	Ave.
Pavllo [27]	3.0	3.1	2.2	3.4	2.3	2.7	2.7	3.1	2.1	2.9	2.3	2.4	3.7	3.1	2.8	2.8
Lin [15]	2.7	2.8	2.1	3.1	2.0	2.5	2.5	2.9	1.8	2.6	2.1	2.3	3.7	2.7	3.1	2.7
UGCN(CPN)*	3.5	3.6	3.0	3.9	3.0	3.4	3.2	3.6	2.9	3.7	3.0	3.1	4.2	3.4	3.7	3.4
UGCN(CPN)	2.3	2.5	2.0	2.7	2.0	2.3	2.2	2.5	1.8	2.7	1.9	2.0	3.1	2.2	2.5	2.3
UGCN(GT)	1.2	1.3	1.1	1.4	1.1	1.4	1.2	1.4	1.0	1.3	1.0	1.1	1.7	1.3	1.4	1.4

4.4 Comparison with state-of-the-art

Results on Human3.6M In this section, we compare the proposed approach to several *state-of-the-art* algorithms in monocular 3D pose estimation from an agnostic camera on Human3.6M dataset. We trained our model on 2D poses predicted by cascaded pyramid network (CPN) [3]. It is the most typical 2D estimator used in previous works. The results on two protocols are shown in the Table 6. As shown in the table, our method achieves promising results on Human3.6 under two metrics(45.6 MPJPE on *Protocol 1* and 35.5 P-MPJPE on *Protocol 2*) which surpass all other baselines. We also examine the result on a more powerful 2D pose estimator HR-Net [31]. It further brings roughly $3mm$ MPJPE improvement.Besides, we also compare our method with others based on ground-truth 2D pose. Details are illustrated in the supplementary materials.

Table 8. Comparison with previous work on the MPI-INF-3DHP dataset. The bold-faced numbers represent the best, while underlined numbers represent the second best.

Method	PCK[↑]	AUC[↑]	MPJPE(mm)[↓]
Mehta [20]	75.7	39.3	-
Mehta(ResNet=50) [21]	77.8	41.0	-
Mehta(ResNet=101) [21]	79.4	41.6	-
Lin(F=25) [15]	83.6	51.4	79.8
Lin(F=50) [15]	82.4	49.6	81.9
UGCN w/o Motion Loss	<u>84.2</u>	<u>54.2</u>	<u>76.7</u>
UGCN	**86.9**	**62.1**	**68.1**

Following [27], we evaluate the dynamic quality of predicted 3D pose sequences by Mean per Joint Velocity Error(MPJVE). This metric measures the smoothness of predicted pose sequences. As shown in Table 7, with motion loss, our method significantly reduces the MPJVE by 32% (from $3.4mm$ to $2.3mm$) and outperforms other baselines.

Results on MPI-INF-3DHP We compare the results of PCK, AUC, and MPJPE against the other state-of-the-art methods on MPI-INF-3DHP dataset with the input of ground-truth 2d skeleton sequences. As shown in Table 8, our approach achieves a significant improvement against other methods. Our method finally achieves 86.9 PCK, 62.1 AUC and 68.1 MPJPE on this dataset. The proposed motion loss significantly improves the accuracy and reduces the error.

Visualization results The qualitative results on Human3.6M and MPI-INF-3DHP are shown in Figure 5. We choose samples with huge movements and hard actions to show the effectiveness of our system. More visualization results comparing with other previous works can be find in the supplementary materials.

5 Conclusion

In this work, we propose a novel objective function, motion loss. It explicitly involves motion modeling into learning. To better optimize model under the supervision of motion loss, the 3D pose estimation should have a long-term perception of pose sequences. It motivated us to design a U-shaped model to capture both short-term and long-term temporal dependencies. On two large datasets, the proposed UGCN with motion loss achieves state-of-the-art performance. The motion loss may inspire other skeleton-based tasks such as action forecasting, action generation and pose tracking.

Acknowledgment This work is partially supported by the SenseTime Collaborative Grant on Large-scale Multi-modality Analysis (CUHK Agreement No. TS1610626 and No.TS1712093), the General Research Fund (GRF) of Hong Kong (No.14236516 and No.14203518).

References

1. Atwood, J., Towsley, D.: Diffusion-convolutional neural networks. In: Advances in Neural Information Processing Systems. pp. 1993–2001 (2016)
2. Cai, Y., Ge, L., Liu, J., Cai, J., Cham, T.J., Yuan, J., Thalmann, N.M.: Exploiting spatial-temporal relationships for 3d pose estimation via graph convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2272–2281 (2019)
3. Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., Sun, J.: Cascaded pyramid network for multi-person pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7103–7112 (2018)
4. Cheng, Y., Yang, B., Wang, B., Yan, W., Tan, R.T.: Occlusion-aware networks for 3d human pose estimation in video. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 723–732 (2019)
5. Dabral, R., Mundhada, A., Kusupati, U., Afaque, S., Sharma, A., Jain, A.: Learning 3d human pose from structure and motion. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 668–683 (2018)
6. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: Advances in neural information processing systems. pp. 3844–3852 (2016)
7. Fang, H.S., Xu, Y., Wang, W., Liu, X., Zhu, S.C.: Learning pose grammar to encode human body configuration for 3d pose estimation. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
8. Ionescu, C., Carreira, J., Sminchisescu, C.: Iterated second-order label sensitive pooling for 3d human pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1661–1668 (2014)
9. Ionescu, C., Papava, D., Olaru, V., Sminchisescu, C.: 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)
10. Ionescu, C., Papava, D., Olaru, V., Sminchisescu, C.: 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 (jul 2014)
11. Iskakov, K., Burkov, E., Lempitsky, V., Malkov, Y.: Learnable triangulation of human pose. arXiv preprint arXiv:1905.05754 (2019)
12. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
13. Lee, K., Lee, I., Lee, S.: Propagating lstm: 3d pose estimation based on joint interdependency. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 119–135 (2018)
14. Li, S., Chan, A.B.: 3d human pose estimation from monocular images with deep convolutional neural network. In: Asian Conference on Computer Vision. pp. 332–347. Springer (2014)
15. Lin, J., Lee, G.H.: Trajectory space factorization for deep video-based 3d human pose estimation. arXiv preprint arXiv:1908.08289 (2019)
16. Lin, M., Lin, L., Liang, X., Wang, K., Cheng, H.: Recurrent 3d pose sequence machines. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 810–819 (2017)

17. Martinez, J., Hossain, R., Romero, J., Little, J.J.: A simple yet effective baseline for 3d human pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2640–2649 (2017)
18. Mehta, D., Rhodin, H., Casas, D., Fua, P., Sotnychenko, O., Xu, W., Theobalt, C.: Monocular 3d human pose estimation in the wild using improved cnn supervision. In: 3D Vision (3DV), 2017 Fifth International Conference on. IEEE (2017). https://doi.org/10.1109/3dv.2017.00064
19. Mehta, D., Rhodin, H., Casas, D., Fua, P., Sotnychenko, O., Xu, W., Theobalt, C.: Monocular 3d human pose estimation in the wild using improved cnn supervision. In: 3D Vision (3DV), 2017 Fifth International Conference on. IEEE (2017). https://doi.org/10.1109/3dv.2017.00064
20. Mehta, D., Rhodin, H., Casas, D., Fua, P., Sotnychenko, O., Xu, W., Theobalt, C.: Monocular 3d human pose estimation in the wild using improved cnn supervision. In: 2017 International Conference on 3D Vision (3DV). pp. 506–516. IEEE (2017)
21. Mehta, D., Sridhar, S., Sotnychenko, O., Rhodin, H., Shafiei, M., Seidel, H.P., Xu, W., Casas, D., Theobalt, C.: Vnect: Real-time 3d human pose estimation with a single rgb camera. ACM Transactions on Graphics (TOG) **36**(4), 44 (2017)
22. Niepert, M., Ahmed, M., Kutzkov, K.: Learning convolutional neural networks for graphs. In: International conference on machine learning. pp. 2014–2023 (2016)
23. Ning, G., Huang, H.: Lighttrack: A generic framework for online top-down human pose tracking. arXiv preprint arXiv:1905.02822 (2019)
24. Park, S., Hwang, J., Kwak, N.: 3d human pose estimation using convolutional neural networks with 2d pose information. In: European Conference on Computer Vision. pp. 156–169. Springer (2016)
25. Pavlakos, G., Zhou, X., Daniilidis, K.: Ordinal depth supervision for 3d human pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7307–7316 (2018)
26. Pavlakos, G., Zhou, X., Derpanis, K.G., Daniilidis, K.: Coarse-to-fine volumetric prediction for single-image 3d human pose. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7025–7034 (2017)
27. Pavllo, D., Feichtenhofer, C., Grangier, D., Auli, M.: 3d human pose estimation in video with temporal convolutions and semi-supervised training. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7753–7762 (2019)
28. Qiu, H., Wang, C., Wang, J., Wang, N., Zeng, W.: Cross view fusion for 3d human pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4342–4351 (2019)
29. Ramakrishna, V., Kanade, T., Sheikh, Y.: Reconstructing 3d human pose from 2d image landmarks. In: European Conference on Computer Vision. pp. 573–586. Springer (2012)
30. Rayat Imtiaz Hossain, M., Little, J.J.: Exploiting temporal information for 3d human pose estimation. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 68–84 (2018)
31. Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
32. Sun, X., Shang, J., Liang, S., Wei, Y.: Compositional human pose regression. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2602–2611 (2017)

33. Tekin, B., Katircioglu, I., Salzmann, M., Lepetit, V., Fua, P.: Structured prediction of 3d human pose with deep neural networks. arXiv preprint arXiv:1605.05180 (2016)
34. Tekin, B., Márquez-Neila, P., Salzmann, M., Fua, P.: Learning to fuse 2d and 3d image cues for monocular body pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3941–3950 (2017)
35. Tekin, B., Rozantsev, A., Lepetit, V., Fua, P.: Direct prediction of 3d body poses from motion compensated sequences. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 991–1000 (2016)
36. Tome, D., Russell, C., Agapito, L.: Lifting from the deep: Convolutional 3d pose estimation from a single image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2500–2509 (2017)
37. Yan, S., Li, Z., Xiong, Y., Yan, H., Lin, D.: Convolutional sequence generation for skeleton-based action synthesis. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4394–4402 (2019)
38. Yan, S., Xiong, Y., Lin, D.: Spatial temporal graph convolutional networks for skeleton-based action recognition. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
39. Zhou, X., Huang, Q., Sun, X., Xue, X., Wei, Y.: Towards 3d human pose estimation in the wild: a weakly-supervised approach. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 398–407 (2017)
40. Zimmermann, C., Brox, T.: Learning to estimate 3d hand pose from single rgb images. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4903–4911 (2017)