TEMPO: Efficient Multi-View Pose Estimation, Tracking, and Forecasting

Rohan Choudhury  Kris M. Kitani  László A. Jeni
Robotics Institute, Carnegie Mellon University
{rchoudhu, kmkitani}@andrew.cmu.edu  laszlojeni@cmu.edu

Figure 1: We propose TEMPO: TEMporal POse Estimation, a method for multi-view, multi-person pose estimation, tracking and forecasting. TEMPO uses a recurrent architecture to learn a spatiotemporal representation, significantly improving the pose estimation accuracy while preserving speed at inference time. In each view, we show TEMPO’s predicted pose skeletons over 10 frames, colored by tracker identity. Lighter colors correspond to previous frames.

Abstract
Existing volumetric methods for predicting 3D human pose estimation are accurate, but computationally expensive and optimized for single time-step prediction. We present TEMPO, an efficient multi-view pose estimation model that learns a robust spatiotemporal representation, improving pose accuracy while also tracking and forecasting human pose. We significantly reduce computation compared to the state-of-the-art by recurrently computing per-person 2D pose features, fusing both spatial and temporal information into a single representation. In doing so, our model is able to use spatiotemporal context to predict more accurate human poses without sacrificing efficiency. We further use this representation to track human poses over time as well as predict future poses. Finally, we demonstrate that our model is able to generalize across datasets without scene-specific fine-tuning. TEMPO achieves 10% better MPJPE with a 33× improvement in FPS compared to TeseTrack on the challenging CMU Panoptic Studio dataset. Our code and demos are available at https://rccchoudhury.github.io/tempo2023/.

1. Introduction
Estimating the pose of several people from multiple overlapping cameras is a crucial vision problem. Volumetric multi-view methods, which lift 2D image features from each camera view to a feature volume then regress 3D pose, are currently the state of the art [47, 42, 53, 23] in this task. These approaches produce significantly more accurate poses than geometric alternatives, but suffer from two key limitations. First, the most accurate methods use either 3D convolutions [47, 42, 55] or cross-view transformers [49] which are slow and prevent real-time inference. Secondly, most methods are designed for estimating pose at a single timestep and are unable to reason over time, limiting their accuracy and preventing their use for tasks like motion prediction.

We propose TEMPO, a multi-view TEMporal POse estimation method that addresses both of these issues. TEMPO uses temporal context from previous timesteps to produce smoother and more accurate pose estimates. Our model tracks people over time, predicts future pose and runs efficiently, achieving near real-time performance on existing benchmarks. The key insight behind TEMPO, inspired by work in 3D object detection [31, 20], is that recurrently aggregating spatiotemporal context results in powerful learned representations while being computationally efficient. To do this, we decompose the problem into three stages, illustrated in Figure 2.

Given an input RGB video from multiple static, calibrated cameras, at a given timestep $t$ we first detect the locations of each person in the scene by unprojecti-
ing image features from each view to a common 3D volume. We then regress 3D bounding boxes centered on each person, and perform tracking by matching the box centers with the detections from the previous timestep $t - 1$. For each detected person, we compute a spatiotemporal pose representation by recurrently combining features from current and previous timesteps. We then decode the representation into an estimate of the current pose as well as poses at future timesteps. Unlike existing work [47, 53, 42, 55], our method is able to perform temporal tasks like tracking and forecasting without sacrificing efficiency.

We evaluate our method on several pose estimation benchmarks. TEMPO achieves state of the art results on the challenging CMU Panoptic Studio dataset [25] by 10%, and is competitive on the Campus, Shelf and Human3.6M datasets. We additionally collect our own multi-view dataset consisting of highly dynamic scenes, on which TEMPO achieves the best result by a large margin. We show that our model achieves competitive results in pose tracking and evaluate the pose forecasting quality on the CMU Panoptic dataset. Additionally, multi-view pose estimation methods are almost always evaluated on the same dataset they are trained on, leading to results that are specific to certain scenes and camera configurations. We measure our method’s ability to generalize across different datasets and find that our method can transfer without additional fine tuning. To summarize, our key contributions are that:

- We develop the most accurate multi-view, multi-person 3D human pose estimation model. Our model uses temporal context to produce smoother and more accurate poses.
- Our model runs efficiently with no performance degradation.
- Our model tracks and forecasts human pose for every person in the scene.
- We evaluate the generalization of our model across multiple datasets and camera configurations.

2. Related Work

3D Pose Estimation, Tracking and Forecasting

Historically, 3D human pose estimation was limited to the single-person case, with [3, 30, 37, 1] using pictorial structure models to improve over basic triangulation. More recent approaches [19, 39, 23, 7, 26] improve this result by using advanced deep architectures like 3D CNNs and transformers, and others [9, 10] introduce priors on human shape for additional performance. Our method is most similar to [23], which uses 3D CNNs to regress pose directly from a feature volume. We also follow [23, 26] in analyzing our model’s transfer to multiple datasets, extending their qualitative, single-person analysis to a quantitative measurement of performance on several multi-person datasets.

In the multi-person setting, early approaches like [3, 12, 13, 52, 5] associate 2D pose estimates from each view, then fuse the matched 2D poses into 3D. Other methods aimed towards multi-person motion capture use Re-ID features [12, 52], 4D graph cuts [54], plane sweep stereo [32], cross-view graph matching [50], or optimize SMPL parameters [14] to produce 3D poses from 2D pose estimates in each view. These methods can generalize across the data sources, but typically have much less accurate predictions compared to volumetric methods. These first unproject learned 2D image features into a 3D volume and regress pose directly from the 3D features with neural networks. Both [23] and [47] use computationally expensive 3D CNNs for the pose estimation step. Follow up work includes Faster VoxelPose [53] which replaces these 3D CNNs with 2D CNNs for a large speedup, and TesseTrack [42], which uses 4D CNNs to reason over multiple timesteps. Our method combines the best of both: we efficiently incorporate spatiotemporal information with only 2D CNNs and a lightweight recurrent network.

3D Object Detection and Forecasting

Historically, 3D object detection and instance segmentation methods for autonomous driving have led development in using multi-view images. One key similarity to our work is the aggregation of 2D image features into a single 3D volume. While [44, 48] use the same bilinear unprojection strategy as our method, several works [58, 17, 31] propose alternatives such as predicting per-pixel depth. Other works also use temporal information for detection for tracking objects through occlusion and spotting hard-to-see objects: [20, 31, 56] concretely demonstrate the benefits of incorporating spatiotemporal context. In particular, FIERY [20] uses temporal information for future instance prediction and BEVFormer [31] efficiently aggregates temporal information with a recurrent architecture, both of which inspired our method. Furthermore, [24, 16] use per-timestep supervision to track
pixels through occlusion, an idea which we adapt for reasoning about human pose over multiple frames.

3. Method

Our method assumes access to calibrated time-synchronized videos from one or more cameras. At training time, we assume access to \( T \) sets of \( N \) RGB images from different cameras, while at inference time, we have a single set of \( N \) images corresponding to the current timestep. In order to enable TEMPO to transfer to new camera configurations and settings, we compute the dimensions of the space and size of the voxel volume directly from the camera matrices. We set the height of the volume to a constant 2 m, while setting the length and width of the volume to be the bounding box of the camera extrinsics from a top-down view, and center the space at the mean of the camera locations.

3.1. Preliminaries

We briefly review the person detection and pose estimation modules used by VoxelPose \[47\], Tessetrack \[42\], and Faster VoxelPose \[53\] that TEMPO builds upon. We refer the reader to the original papers for further detail.

3.1.1 Person Detection

The detection module aims to estimate the location of the root joint as well as tight 3D bounding box for each person in the scene. Following previous work, we define the root joint as the mid-hip. At a given time \( t \), the detector module takes as input a set of \( N \) images, each corresponding to a different camera view of the same scene at time \( t \). For each image, we extract features with a pretrained backbone, resulting in \( N \) feature maps \( F^t_1, F^t_2, \ldots, F^t_N \).

Given the camera matrices for each view \( C^t_1, C^t_2, \ldots, C^t_N \), we use the bilinear sampling procedure from \[23, 47, 17\]. For a voxel \( v \in V \) with coordinates \( x \), we have

\[
v = \sum_{i=1}^{N} F^t_i(C_i x)
\]

where \( F^t_i(x) \) is the feature map \( F^t_i \)'s value at position \( x \), obtained by bilinear sampling. We then compute a birds-eye view representation of \( V \) by taking the maximum along the \( z \)-axis:

\[
V^t_{BEV} = \max_z V
\]
nally, we regress width, length and centerness from \( \mathbf{F}_{\text{BEV}} \) with a multi-headed 2D CNN to produce bounding box predictions for each proposal. The loss function for the detection module has three terms. First, \( L_{2D} \) is the distance between the predicted 2D heatmap and the ground truth, given by

\[
L_{2D} = \sum_{t=0}^{T} \sum_{(x,y)} \| \mathbf{H}(x,y) - \mathbf{H}_{\text{GT}}(x,y) \|_2
\]

(3)

We also compute the loss on the predicted 1D heatmap:

\[
L_{1D} = \sum_{t=k=1}^{T} \sum_{z} \| \mathbf{H}_k(z) - \mathbf{H}_{k,\text{GT}}(z) \|_2
\]

(4)

Finally, we include the bounding box regression loss

\[
L_{\text{bbox}} = \sum_{t=1}^{T} \sum_{(i,j) \in U} \| \mathbf{S}(i,j) - \mathbf{S}_{\text{GT}}(i,j) \|_1
\]

(5)

The total detection loss is the sum of the above terms, with

\[
L_{\text{det}} = L_{2D} + L_{1D} + L_{\text{bbox}}.
\]

### 3.2. Person Tracking

We now describe how TEMPO uses temporal information. Unlike previous works, TEMPO takes as input a set of \( N \) images at time \( t \) as well as the person detections \( D_{t-1} \) and corresponding 2D pose embeddings \( \mathbf{P}^{t-1} \) from the previous timestep.

Each proposed detection from the previous step consists of a body center \( c_i \) and a bounding box \( B^i = (h^i, w^i) \). Given \( K \) proposals, we compute a \( K \times K \) cost matrix \( \mathbf{A} \) based on the IoU between \( B^i \) and \( B^j \) for all detections at time \( t \), resulting in

\[
\mathbf{A}[i,j] = \| c_i - c_j \|
\]

(7)

with \( c_i, c_j \) being the associated predicted locations of person \( i \) and person \( j \)’s root joint.

While VoxelTrack \([55]\) computes cost across every joint in each pose, TEMPO uses the top-down view of the associated bounding box for each person. At inference time, we use the SORT \([42]\) tracker, which is fast and uses a simple Kalman filter with no learned Re-ID mechanism. While \([42]\) uses a learned tracker based on SuperGlue \([43]\), we find that SORT is faster and does not result in degraded performance.

### 3.3. Temporal Pose Estimation and Forecasting

After running the tracker, the input to the pose estimation stage is a set of detection proposals \( D^t \), the previous timestep’s detection proposals \( D^{t-1} \), and the assignment matrix between the detection sets \( \mathbf{A}^t \). We also assume access to pose features \( \mathbf{P}^{t-1} \) for each person in the previous timestep. Both \( D^t \) and \( D^{t-1} \) have \( K \) proposals each.

However, \( \mathbf{P}^{t-1}_i \) and \( \mathbf{P}^t_i \) are centered at \( c^t_i \) and \( c^{t-1}_i \) respectively, and thus the pose features from each volume are not in the same coordinate system due to the motion of person \( i \). To fix this, we follow the standard procedure used in temporal birds-eye view prediction \([31, 20]\) and warp the projected features from \( \mathbf{P}^{t-1}_i \) into the coordinate system of \( \mathbf{P}^t_i \) with a translational warp defined by \( c^t_i - c^{t-1}_i \).

After warping the previous pose features, we run a recurrent network with Spatial Gated Recurrent Units \([2]\) (SpatialGRUs) to produce multiple embeddings: \( \mathbf{P}^t_i \), representing the current pose, and \( \mathbf{P}^{t+1}_i, \mathbf{P}^{t+2}_i, \ldots \), representing the pose in future frames. While \([20]\) and \([31]\) do not propagate gradients through time and only predict object locations and instance segments at time \( T \), we do backpropagate through time by predicting the pose for each person at every timestep. At training time, we recurrently compute the temporal representation at each timestep \( t_0, t_0 + 1, \ldots, t_0 + T \), decode a pose for every timestep, and compute losses over all the predicted poses simultaneously. Thus, the final train-
Figure 3: A closer look at the temporal representation used by our model. Following [53], we first project the feature volume to each of the three planes, and concatenate the projections channel-wise. We pass this feature map through an encoder network. We use this feature encoding as input to the SpatialGRU module, using the spatially warped pose feature from the previous timestep as a hidden state. We use the SpatialGRU module to produce features at the current and future timesteps, which we decode into human poses with the pose decoder network.

The training objective is

\[ L_{\text{pose}} = \sum_{t=1}^{T} \sum_{i=1}^{K} L_{\text{joint},i,t} + L_{\text{joint},i,t+1} \]  

where \( L_{\text{joint},i,t} \) is the L1 distance between the predicted and ground truth pose at time \( t \). Providing supervision to the network at every timestep allows the network to learn a representation that encodes the motion between consecutive frames while enabling temporal smoothness between predictions. As we show in Section 4.5, this technique is crucial to our model’s performance.

While training, we run the network \( T \) times, for each input timestep. However, at inference time, we save the previous embeddings and only receive input images at a single timestep \( t \), significantly reducing the computational burden and allowing our model to use temporal information without sacrificing efficiency.

4. Experiments

4.1. Datasets and Metrics

**Panoptic Studio** The CMU Panoptic Studio dataset [25] is a large multi-view pose estimation dataset with several synchronized camera sequences of multiple interacting subjects. Following prior work [47][53][49][52], we use five HD cameras, specifically cameras 3, 6, 12, 13, and 23. We also use the same training and test split as these works, omitting the sequence 160906_band3 due to data corruption.

**Human 3.6M** The Human3.6M dataset [22][21] consists of videos of a single subject in an indoor studio with four static cameras. Each video has a professional actor performing a specific action. We follow the training-test split of prior works, using subjects 9 and 11 for validation and the others for training, while omitting corrupted sequences.

**Campus and Shelf** The Campus and Shelf datasets [3] contain approximately 4000 frames of a single scene. While these datasets are commonly used for benchmarking in previous work, they are missing many annotations. We follow previous work [47][53][49] and adopt a synthetic heatmap-based scheme for training.

**EgoHumans** We include the newly collected EgoHumans multi-view dataset [28]. This benchmark consists of approximately 1 hour of video of up to 5 subjects performing highly dynamic activities, such as playing tag, fencing, or group assembly. It contains videos from up to eight fisheye cameras and includes both egocentric and exocentric camera and pose data.

**Metrics** For the Panoptic Studio, Human3.6M, and EgoHumans datasets, we report the mean per-joint position error (MPJPE). We additionally report the Average Precision (AP) on the Panoptic and EgoHumans datasets and report MPJPE for Human3.6M in line with previous work [37][35][23][42]. On the Shelf and Campus dataset we report the Percentage of Correct Parts (PCP3D). For pose forecasting, we measure the MPJPE between the predicted pose and the ground truth pose 0.33s into the future, matching previous work [56].
Table 1: Pose estimation results on the CMU Panoptic dataset. Our method achieves the best MPJPE and AP while running at speed comparable to Faster VoxelPose. We evaluate our methods at $384 \times 384$ resolution on the Panoptic dataset, as well as the higher resolution used in other methods. We mark TesseTrack [42] with a † as their reported results are on a different data split, and the results in this table are from our best reproduction of the method, which is not public.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>Resolution</th>
<th>AP$_{25}$ ↑</th>
<th>AP$_{50}$ ↑</th>
<th>AP$_{100}$ ↑</th>
<th>AP$_{150}$ ↑</th>
<th>MPJPE (mm) ↓</th>
<th>FPS (s)↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>VoxelPose [47]</td>
<td>ResNet-50</td>
<td>960 × 512</td>
<td>83.59</td>
<td>98.33</td>
<td>99.76</td>
<td>99.91</td>
<td>17.68</td>
<td>3.2</td>
</tr>
<tr>
<td>Faster VoxelPose [53]</td>
<td>ResNet-50</td>
<td>960 × 512</td>
<td>85.22</td>
<td>98.08</td>
<td>99.32</td>
<td>99.48</td>
<td>18.26</td>
<td>31.1</td>
</tr>
<tr>
<td>MvP [32]</td>
<td>ResNet-50</td>
<td>960 × 512</td>
<td>92.28</td>
<td>96.6</td>
<td>97.45</td>
<td>97.69</td>
<td>15.76</td>
<td>3.6</td>
</tr>
<tr>
<td>Ours</td>
<td>ResNet-50</td>
<td>960 × 512</td>
<td>89.01</td>
<td>99.08</td>
<td>99.76</td>
<td>99.93</td>
<td>14.68</td>
<td>29.3</td>
</tr>
<tr>
<td>VoxelPose [47]</td>
<td>HRNet</td>
<td>384 × 384</td>
<td>82.44</td>
<td>98.55</td>
<td>99.74</td>
<td>99.92</td>
<td>17.63</td>
<td>2.3</td>
</tr>
<tr>
<td>Faster VoxelPose [53]</td>
<td>HRNet</td>
<td>384 × 384</td>
<td>81.69</td>
<td>98.38</td>
<td>97.39</td>
<td>97.89</td>
<td>16.34</td>
<td>2.8</td>
</tr>
<tr>
<td>PlaneSweepPose [32]</td>
<td>HRNet</td>
<td>384 × 384</td>
<td>90.41</td>
<td>96.32</td>
<td>97.89</td>
<td>98.4</td>
<td>16.92</td>
<td>0.6</td>
</tr>
<tr>
<td>MvP [49]</td>
<td>HRNet</td>
<td>384 × 384</td>
<td>99.3</td>
<td>95.1</td>
<td>97.8</td>
<td>97.7</td>
<td>95.5</td>
<td>97.9</td>
</tr>
<tr>
<td>TEMPO (Ours)</td>
<td>HRNet</td>
<td>384 × 384</td>
<td>89.32</td>
<td>98.48</td>
<td>99.73</td>
<td>99.94</td>
<td>15.99</td>
<td>20.3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>Resolution</th>
<th>PCP3D Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Belagiannis et al. [3]</td>
<td></td>
<td></td>
<td>66.1 ± 1.0</td>
</tr>
<tr>
<td>Ershadi et al. [15]</td>
<td></td>
<td></td>
<td>93.3 ± 1.0</td>
</tr>
<tr>
<td>Dong et al. [12]</td>
<td></td>
<td></td>
<td>98.8 ± 0.1</td>
</tr>
<tr>
<td>VoxelPose [47]</td>
<td></td>
<td></td>
<td>99.3 ± 0.1</td>
</tr>
<tr>
<td>Faster VoxelPose [53]</td>
<td></td>
<td></td>
<td>99.4 ± 0.1</td>
</tr>
<tr>
<td>PlaneSweepPose [32]</td>
<td></td>
<td></td>
<td>99.3 ± 0.1</td>
</tr>
<tr>
<td>MvP [49]</td>
<td></td>
<td></td>
<td>99.3 ± 0.1</td>
</tr>
<tr>
<td>TEMPO (Ours)</td>
<td>HRNet</td>
<td>384 × 384</td>
<td>99.0 ± 0.1</td>
</tr>
</tbody>
</table>

4.2. Implementation Details
Following [47, 53, 56, 49] we use a ResNet-50 backbone pre-trained on the Panoptic dataset and follow [23] by also using a ResNet-50 backbone pre-trained on Human3.6M for the dataset-specific pose estimation results. We use HRNet [45] pre-trained on COCO as the model backbone for the generalization experiments, as done in [32] rather than pose estimation backbones that are trained on multi-view datasets. All methods are trained on 8 NVIDIA A100 GPUs with batch size of 2 per GPU. We use Adam with a learning rate of 3e-4, with weight decay of 1e-4 and a linear decay schedule for 10 epochs. We measure FPS using a single A100 GPU, and our code is based off the MMPose [11] library. Additional architectural details are in the supplement.

4.3. Pose Estimation Results
We first compare the results of our methods other state-of-the-art methods. On the Panoptic Studio dataset, we report results following [53, 47, 49] and use $960 \times 512$ images and initialize from a ResNet-50 [18, 51] checkpoint pretrained on the Panoptic dataset. We also evaluate our method using an HRNet [45] backbone pretrained on COCO with $384 \times 384$ images for a fair comparison with TesseTrack. In Table 1 we provide a complete comparison across the state-of-the-art methods, with baselines trained on the same image resolutions and backbones for completeness. TEMPO achieves significantly lower MPJPE across both resolutions and backbones, while running at 29.3 FPS, competitive with Faster VoxelPose. We attribute this performance increase to the the smoother and more temporally consistent skeletons our model produces due to its incorporation of temporal context and temporal supervision. We also show in Table 2 that our model achieves performance competitive with the state of the art on the Campus and Shelf datasets.
We compare pose estimation performance on the Human3.6M and EgoHumans datasets in Table 4. Our results perform comparably to the state-of-the-art. Notably, \cite{23} uses ground-truth locations and cropped bounding boxes from input views, while our method is able to match performance despite simultaneously detecting people in the scene. Furthermore, our method significantly outperforms others in the more challenging EgoHumans benchmark, suggesting that temporal context is crucial for handling rapid motion.

### 4.4. Pose Tracking and Forecasting

We compared the performance of TEMPO’s tracking to VoxelTrack \cite{55} in Table \ref{tab:tracking}. Our tracker is competitive but performs slightly worse, which is expected due to its lack of learned Re-ID features. To our knowledge, pose forecasting has not been attempted for the multi-view case, so we compare against the closest forecasting method, Snipper \cite{56}, which estimates future pose from monocular video. Our method takes as input 4 previous timesteps and predicts the pose over the next 3 timesteps, which is 0.33 seconds into the future, matching prior work \cite{56}. Our model achieves state-of-the-art performance, shown in Table \ref{tab:tracking}.

We conducted an additional experiment to measure the performance of our model in transfer on different datasets. The standard practice in monocular 3D pose estimation is to train on and evaluate performance on a combination of multiple datasets. However, the predominant paradigm for

<table>
<thead>
<tr>
<th>Method</th>
<th>MOTA</th>
<th>IDF1</th>
<th>MPJPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>VoxelTrack \cite{55}</td>
<td>98.45</td>
<td>98.67</td>
<td>-</td>
</tr>
<tr>
<td>Snipper \cite{56}</td>
<td>93.40</td>
<td>85.50</td>
<td>40.2</td>
</tr>
<tr>
<td>TEMPO (Ours)</td>
<td>98.42</td>
<td>93.62</td>
<td>38.5</td>
</tr>
</tbody>
</table>

(a) Evaluation of tracking and forecasting on the CMU Panoptic dataset. Our method outperforms \cite{56} by using multi-view information and is competitive with VoxelTrack. \cite{55}

<table>
<thead>
<tr>
<th>Training Dataset</th>
<th>Panoptic</th>
<th>H.6M</th>
<th>EgoHumans</th>
<th>Panoptic</th>
<th>H.6M</th>
<th>EgoHumans</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>14.68</td>
<td>62.96</td>
<td>119.8</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>36.56</td>
<td>25.3</td>
<td>108.3</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>48.9</td>
<td>48.9</td>
<td></td>
</tr>
</tbody>
</table>

(b) Evaluating TEMPO’s ability to transfer. Unlike previous methods, TEMPO is able to train on multiple datasets and can perform reasonably given enough training data. While our model is able to effectively transfer to Human3.6M, it has difficulty with EgoHumans due likely due to its use of fisheye cameras.

Figure 4: Samples of our model’s pose estimation performance on the Panoptic Studio, Human3.6M, and EgoHumans datasets. TEMPO predicts accurate poses and tracks them over time.
Table 4: Pose Estimation results on the Human3.6M and EgoHumans datasets in MPJPE (mm). Our method is competitive with the state-of-the-art on Human3.6M and surpasses current methods by a significant margin on EgoHumans.

<table>
<thead>
<tr>
<th>Method</th>
<th>Human3.6M</th>
<th>EgoHumans</th>
</tr>
</thead>
<tbody>
<tr>
<td>Martinez et al. [35]</td>
<td>57.0</td>
<td>-</td>
</tr>
<tr>
<td>Pavlakos et al. [37]</td>
<td>56.9</td>
<td>-</td>
</tr>
<tr>
<td>Kadkhodamohammadi et al. [26]</td>
<td>49.1</td>
<td>-</td>
</tr>
<tr>
<td>Ma et al. [33]</td>
<td>24.4</td>
<td>-</td>
</tr>
<tr>
<td>VoxelPose [47]</td>
<td>19.0</td>
<td>46.24</td>
</tr>
<tr>
<td>Faster VoxelPose [53]</td>
<td>19.8</td>
<td>48.50</td>
</tr>
<tr>
<td>MvP [49]</td>
<td>18.6</td>
<td>41.72</td>
</tr>
<tr>
<td>Iskakov et al. [23]</td>
<td>17.7</td>
<td>-</td>
</tr>
<tr>
<td>TEMPO (Ours)</td>
<td>18.5</td>
<td>36.74</td>
</tr>
</tbody>
</table>

Table 5: Ablation study on various components to our model. The most important component to performance was the per-timestep supervision. Warping the previous feature also improved performance. We observed the forecasting and slightly increasing the length of the input history had no noticeable effect on performance.

<table>
<thead>
<tr>
<th>Method</th>
<th>Forecasting</th>
<th>Warping</th>
<th>Per-t</th>
<th>loss</th>
<th>MPJPE ↓(mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td></td>
<td></td>
<td>3</td>
<td>✓</td>
<td>17.83</td>
</tr>
<tr>
<td>b</td>
<td></td>
<td>✓</td>
<td>3</td>
<td>✓</td>
<td>15.03</td>
</tr>
<tr>
<td>c</td>
<td></td>
<td>✓ ✓</td>
<td>3</td>
<td>✓</td>
<td>14.94</td>
</tr>
<tr>
<td>d</td>
<td></td>
<td>✓ ✓ ✓</td>
<td>3</td>
<td>✓</td>
<td>14.68</td>
</tr>
<tr>
<td>e</td>
<td></td>
<td>✓ ✓ ✓</td>
<td>4</td>
<td>✓</td>
<td>14.90</td>
</tr>
<tr>
<td>f</td>
<td></td>
<td>✓ ✓ ✓</td>
<td>5</td>
<td>✓</td>
<td>14.82</td>
</tr>
</tbody>
</table>

Table 6: Ablation on number of cameras. We observe that performance decreases with decreasing number of cameras.

<table>
<thead>
<tr>
<th>Cameras</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPJPE</td>
<td>14.68</td>
<td>17.34</td>
<td>19.22</td>
<td>32.13</td>
<td>51.32</td>
</tr>
</tbody>
</table>

4.5. Ablations

We train ablated models to study the impact of individual components in our method. All our experiments are conducted on the CMU Panoptic dataset, with results shown in Table 5. We find that using temporal information slightly helps, but with per-timestep supervision, the model is able to greatly improve. Warping the pose between timesteps further improves the model. We hypothesize that the improvement from warping is small due to the relative lack of motion in the Panoptic dataset - the distance between body centers in consecutive timesteps is usually small. We also measure the effect of the history length on performance and found no significant difference. While intuitively, larger history lengths should provide more context, the GPU memory constraints of our method prevent investigating $T > 5$.

We also ablated on the number of cameras on the Panoptic dataset. We found that the MPJPE increases with the number of cameras, matching the findings of [47, 23] and [42].

5. Conclusions

Understanding human behavior from video is a fundamentally temporal problem, requiring accurate and efficient pose estimation algorithms that can reason over time. We presented the first method that satisfies these requirements, achieving state-of-the-art results over existing multi-view pose estimation benchmarks via temporal consistency as a learning objective. Our model is also highly efficient, relying on recurrence to maintain a temporal state while enabling pose tracking and forecasting. TEMPO represents a step closer towards general-purpose human behavior understanding from video.

Acknowledgements

This research was supported partially by Fujitsu.
References


