Efficient Online Multi-Person 2D Pose Tracking with Recurrent Spatio-Temporal Affinity Fields

Yaadhav Raaj Haroon Idrees Gines Hidalgo Yaser Sheikh
The Robotics Institute, Carnegie Mellon University
{raaj@cmu.edu, hidrees@andrew.cmu.edu, gines@cmu.edu, yaser@cs.cmu.edu}

Abstract

We present an online approach to efficiently and simultaneously detect and track the 2D pose of multiple people in a video sequence. We build upon Part Affinity Field (PAF) representation designed for static images, and propose an architecture that can encode and predict Spatio-Temporal Affinity Fields (STAF) across a video sequence. In particular, we propose a novel temporal topology cross-linked across limbs which can consistently handle body motions of a wide range of magnitudes. Additionally, we make the overall approach recurrent in nature, where the network ingests STAF heatmaps from previous frames and estimates those for the current frame. Our approach uses only online inference and tracking, and is currently the fastest and the most accurate bottom-up approach that is runtime invariant to the number of people in the scene and accuracy invariant to input frame rate of camera. Running at ∼30 fps on a single GPU at single scale, it achieves highly competitive results on the PoseTrack benchmarks. [1]

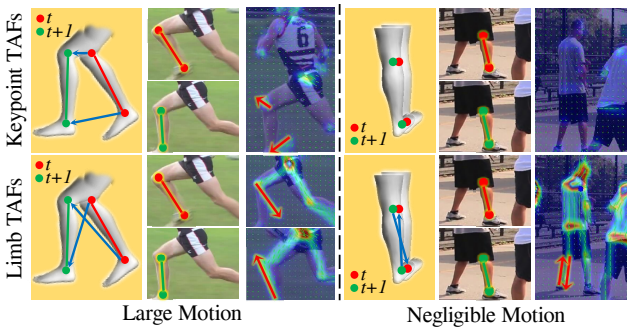

Figure 1: We solve multi-person human pose tracking by encoding change in position and orientation of keypoints or limbs across time as Temporal Affinity Fields (TAFs) in a recurrent fashion. **Top:** Modeling TAFs (blue arrows) through keypoints works when motion occurs but fails during limited motion making temporal association difficult. **Bottom:** Cross-linked TAFs across limbs perform consistently for all kinds of motions providing redundancy and smoother encoding for further refinement and prediction.

1. Introduction

Multi-person human pose estimation has received considerable attention in the past few years assisted by deep convolutional learning as well as COCO [20] and MPII [3] datasets. The recently introduced PoseTrack dataset [16] has provided the community with a large scale corpus of video data with multiple people in the scenes. In this paper, our aim is to utilize these towards building a truly online and real-time multi-person 2D pose estimator and tracker that is deployable and scalable while achieving high performance and requiring minimal post-processing, with potential uses in real-time and closed-loop applications with low latency where the execution is in sync with frame rate of camera such as self-driving cars and augmented reality.

The real-time/online nature of such an approach introduces several challenges: i) scenes with multiple people demand handling of occlusion, proximity and contact as well as limb articulation, and ii) it should be run-time invariant to the number of people in the scene. Furthermore, iii) it must be capable of handling challenges induced from video data, such as large camera motion and motion blur across frames. We build upon the Part Affinity Fields (PAFs) [5] to overcome these challenges, which represent connections across body keypoints in static images as normalized 2D vector fields with position and orientation. In this work, we propose Temporal Affinity Fields (TAFs) which encode connections between keypoints across frames, including a unique cross-linked limb topology as seen in bottom row of Figure 1. In the absence of motion or when there is not enough data from previous frames, TAFs constructed between same keypoints, e.g., wrist-wrist or elbow-elbow across frames lose all associative properties (see top row of Fig. 1). In this case, the nullification of magnitude and orientation provides no useful information to discern between

[1] Video

the case where a new person appears or where an existing person stops moving. This effect is compounded if these two cases occur in proximity together. However, the longer limb TAF connections allow information preservation even in the absence of motion or appearance of new people by preventing corruption of valid information with noise as the magnitude of motion becomes small. In the limiting case of zero motion, *the TAF effectively collapses to a PAF*. From the perspective of a network, TAF between keypoints destroys spatial information about keypoints as motion ceases, whereas TAF across keypoints simply learns to propagate the PAF, which is a much simpler task.

Furthermore, we work on videos in a recurrent manner to make the approach real-time, where computation of each frame leverages information from previous frames thereby reducing overall computation. Where the single-image pose estimation methods use multiple stages to refine heatmaps [5, 25], we exploit the redundant information in the video frames and divert the resources towards efficient computation of both poses and tracking across multiple frames. Thus, the multi-stage computation over images is divided over multiple frames in a video. Overall, we call this Recurrent Spatio-Temporal Affinity Fields (STAF) and it achieves highly competitive results on the PoseTrack benchmarks: [64.6% mAP, 58.4% MOTA] on single scale at $\sim$30 FPS, and [71.5% mAP, 61.3% MOTA] on multiple scales at $\sim$7 FPS on the Posetrack 2017 validation set using one GTX 1080 Ti. Our approach currently ranks second place for accuracy and third place for tracking on the 2017 challenge [1]. Note that, our tracking approach is truly online on a per-frame basis with no post improvements made to tracks.

The rest of the paper is organized as follows. In Sec. 2, we discuss related work and situate the paper in the literature. In Sec. 3 we present details of our approach, training procedure as well as tracking and inference algorithm. Finally we present results and ablation experiments in Sec. 8 and conclude the paper in Sec. 5.

2. Related Work

Early methods for human pose estimation localized keypoints or body parts of individuals but did not consider multiple people simultaneously [4, 29, 39, 19, 35]. Hence, these methods were not adept at localizing keypoints of highly articulated or interacting people. Person detection was used followed by single-person keypoint detection [30, 10, 34, 15]. With deep learning, human detection methods such as Mask-RCNN [9, 12] were employed to directly predict multiple human bounding boxes through ROI-pooling followed by pose estimation per person [31]. However, these methods suffered when people were in close proximity as bounding boxes got grouped together. Furthermore, these **top-down** methods required more computation as the number of people increased in the image, making them inadequate for real-time pose estimation and tracking.

The **bottom-up** Part Affinity Fields (PAF) method [5] produced a spatial encoding of pair-wise body part connections in image space, followed by greedy bipartite graph matching for inference permitting consistent speed irrespective of the number of people. Person Lab [26] built upon these ideas to incorporate redundant connections on people with a less greedy inference approach getting highly competitive results on the COCO [21] and MPII [3] datasets. These methods work on single images and do not incorporate any keypoint tracking or past information.

Many offline methods have been used to enforce temporal consistency of pose estimation in videos [14, 16, 37]. These require solving spatio-temporal graphs or incorporating data from future frames making them inadequate for an online approach. Alternatively, Song *et al*. and Pfister *et al*. [28, 33] demonstrate how optical flow fields could be predicted per keypoint by formulating the input to be multi-framed. LSTM Pose Machines [23] built upon previous work demonstrating use of single stage/frame for video sequences. However, these networks did not model spatial relationship between keypoints and were evaluated on the single person Penn Action [36] and JHMDB [17] datasets.

A different line of works explored maintaining temporal graphs in neural networks for handling multiple people [8, 7]. Rohit *et al*. demonstrated that a 3D extension of Mask-RCNN, called person tubes, can connect people across time. However, this required applying grouped convolutions over a stack of frames reducing speed, and did not achieve better results for tracking than the Hungarian Algorithm baseline. Joint Flow [7] used the concept of Temporal Flow Field which connected keypoints across two frames. However, it did not use a recurrent structure and explicitly required a pair of images as input increasing run-time significantly. The flow representation also suffered from ambiguity when subjects moved slowly or were stationary and required special handling of such cases during tracking.

Top-down pose and tracking methods [37, 35, 6, 27, 12] have dominated the detection and tracking tasks [37] [38] in PoseTrack but their speed suffered due to explicit human detection and follow-up keypoint detection for each person. Moreover, modeling long-term spatio-temporal graphs for tracking in an offline manner hurt real-time applications. None of these works are able to report any significant runtime to performance measures as they cannot run in real time. In this work, we demonstrate this problem can be solved in a simple elegant single-stage network that incorporates recurrence by using the previous pose heatmaps to predict both keypoints and their spatio-temporal associations. We call this Recurrent Spatio-Temporal Affinity Fields (STAF) which not only represents the prediction of Spatial (PAFs) and Temporal (TAFs) Affinity Fields but also how they are refined through past information.

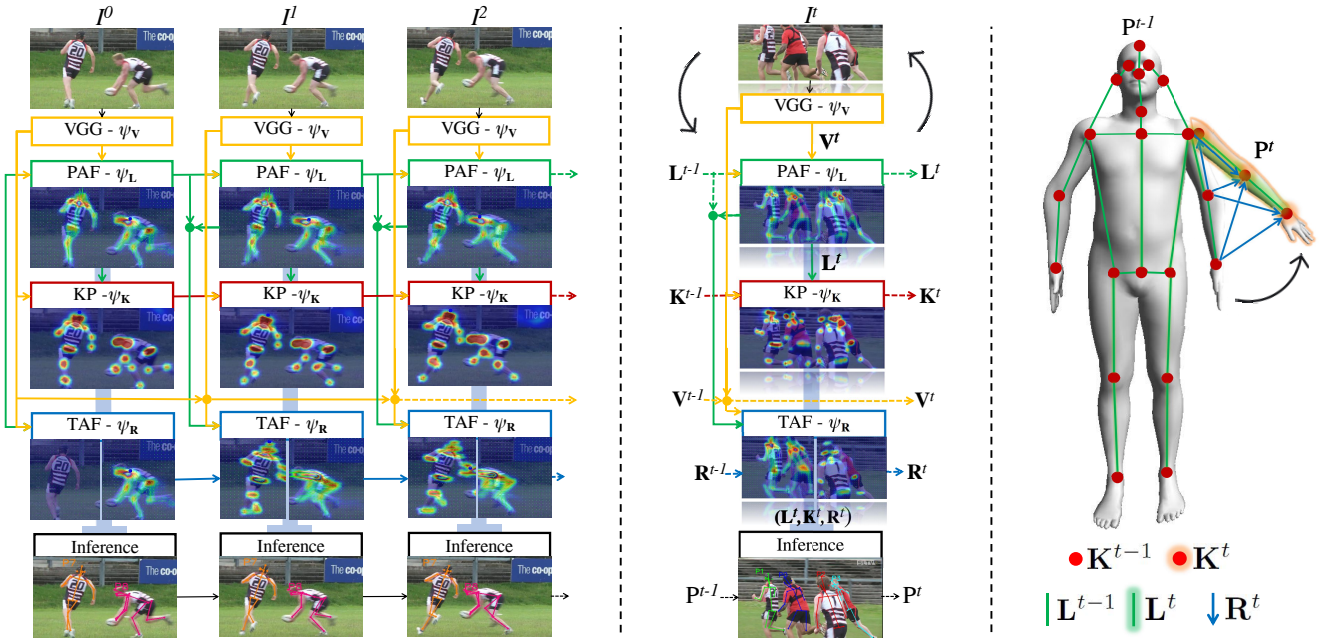

Figure 2: **Left:** Training architecture for one of our models which ingests video sequences in a recurrent manner across time while generating keypoints and connections across keypoints in same frame as Part Affinity Fields (PAFs), and connections across keypoints in time as Temporal Affinity Fields (TAFs). Together, we call this Recurrent Spatio-Temporal Affinity Fields (STAFs). Each module ingests outputs from other modules in both previous and current frames (shown with arrows) and refines it. **Center:** During inference, our network operates on a single video frame at each time step. **Right:** During inference, we use the predicted heatmaps to track and detect people. Keypoints (red) are extracted first, then associated into poses and tracklets using PAFs (green), TAFs (blue) and previous tracklets.

3. Proposed Approach

Our approach aims to solve the problems of keypoint estimation and tracking simultaneously in videos. We employ Recurrent Convolutional Neural Networks which we construct from four essential building blocks. Let $\mathbf{P}^t$ represent the pose of a person in a particular frame or time t, consisting of keypoints $\mathbf{K} = \{\mathbf{K}_1, \mathbf{K}_2, \dots \mathbf{K}_K\}$. The Part Affinity Fields (PAFs) $\mathbf{L} = \{\mathbf{L}_1, \mathbf{L}_2, \dots \mathbf{L}_L\}$ are synthesized from keypoints in each frame. For tracking keypoints across frames a video, we propose Temporal Affinity Fields (TAFs) given by $\mathbf{R} = \{\mathbf{R}_1, \mathbf{R}_2, \dots \mathbf{R}_R\}$ which capture the recurrence and connect the keypoints across frames. Together, they are referred to as Spatio-Temporal Affinity Fields (STAFs). These blocks are visualized in Fig. 2 where each block is shown with a different color: the raw convolutional feature from VGG backbone [32] are shown in amber, PAFs in green, keypoints in red and TAFs in blue.

Thus, the output of VGG backbone, PAFs, keypoints and TAFs are given by $\mathbf{V}$, $\mathbf{L}$, $\mathbf{K}$ and $\mathbf{R}$, respectively, and computed through CNNs by $\psi_\mathbf{V}$, $\psi_\mathbf{L}$, $\psi_\mathbf{K}$ and $\psi_\mathbf{R}$ respectively. The keypoint heatmaps are constructed from ground truth by placing a Gaussian kernel at the location of the annotated keypoint, whereas the PAFs and TAFs are constructed from ground truth between pairs of keypoints for each person:

$$\widetilde{\mathbf{L}}^t_{k\to k'} := \Omega(\widetilde{\mathbf{K}}^t_k, \widetilde{\mathbf{K}}^t_{k'}), \quad \widetilde{\mathbf{R}}^t_{k\to k'} := \Omega(\widetilde{\mathbf{K}}^{t-1}_k, \widetilde{\mathbf{K}}^t_{k'}), \quad (1)$$

where $\sim$ denotes the ground truth and the function $\Omega(\cdot)$ places a directional unit vector at every pixel within a predefined radius of the line connecting the two keypoints.

3.1. Video Models for Pose Estimation and Tracking

Next, we present the three models comprising the four blocks capable of estimating keypoints and STAFs. The input to each network consists of a set of consecutive frames of a video. Each block in each network consists of five 7×7 and two 1×1 convolution layers. Each 7×7 layer is replaceable with the concatenation of three 3×3 convolution layers providing the same receptive field. The first stage has a unique set of weights from subsequent frames as it cannot incorporate any previous data and also has a lower depth which was found to improve results (see Sec. 8). The VGG features are computed for each frame. For frame $\mathbf{I}^t$ at time t of the video, they are computed as $\mathbf{V}^t = \psi_\mathbf{V}(\mathbf{I}^t)$.

Model I: Given $\mathbf{V}^{t-1}$ and $\mathbf{V}^t$, the the following equations

describe the first model:

$$\begin{aligned}\mathbf{L}^t &= \psi_{\mathbf{L}}(\mathbf{V}^t, \psi_{\mathbf{L}}^{q-1}(\cdot)),\\ \mathbf{K}^t &= \psi_{\mathbf{K}}(\mathbf{V}^t,\ \psi_{\mathbf{L}}^{q}(\cdot),\ \psi_{\mathbf{K}}^{q-1}(\cdot)),\\ \mathbf{R}^t &= \psi_{\mathbf{R}}(\mathbf{V}^{t-1},\ \mathbf{V}^t,\ \mathbf{L}^{t-1},\ \mathbf{L}^t,\ \mathbf{R}^{t-1}),\end{aligned} \tag{2}$$

where ψ^q means q recursive applications of ψ. In our experiments, we found that performance plateaus at $q = 5$. In Model I, PAFs are obtained by recursive application of $\psi_{\mathbf{L}}$ on concatenated input from VGG features and PAFs from previous stage. Similarly, keypoints depend on VGG features, keypoints from the previous stage and PAFs from the current stage. Finally, TAFs are dependent on VGG features and PAFs from both the previous and current frames, as well as TAFs from previous frame. This model produces good results but is the slowest due to recursive stages.

Model II: Unlike Model I with multiple applications of CNNs for PAFs and keypoints, Model II computes the PAFs and keypoints in a single pass as visualized in Fig. 2:

$$\begin{aligned}\mathbf{L}^t &= \psi_{\mathbf{L}}(\mathbf{V}^t, \mathbf{L}^{t-1}),\\ \mathbf{K}^t &= \psi_{\mathbf{K}}(\mathbf{V}^t,\ \mathbf{L}^t,\ \mathbf{K}^{t-1}),\\ \mathbf{R}^t &= \psi_{\mathbf{R}}(\mathbf{V}^{t-1},\ \mathbf{V}^t,\ \mathbf{L}^{t-1},\ \mathbf{L}^t,\ \mathbf{R}^{t-1}).\end{aligned} \tag{3}$$

Replacing five stages with a single stage is expected to drop performance. Therefore, the multi-stage computation of PAFs and keypoints in Model II is supplanted with output of PAFs and keypoints from the previous frames. This boosts up the speed significantly without significant loss in performance as it takes advantage of the redundant information in videos, i.e., the PAFs and keypoints from previous frame are a reliable guide to the location of PAFs and keypoints in the current frame. **Model III:** Finally, the third model attempts to estimate Part and Temporal Affinity Fields through a single CNN:

$$\begin{aligned}[\mathbf{L}, \mathbf{R}]^t &= \psi_{[\mathbf{L},\mathbf{R}]}(\mathbf{V}^{t-1}, \mathbf{V}^t, [\mathbf{L}, \mathbf{R}]^{t-1}),\\ \mathbf{K}^t &= \psi_{\mathbf{K}}(\mathbf{V}^t,\ \mathbf{L}^t,\ \mathbf{K}^{t-1}),\end{aligned} \tag{4}$$

where $[\mathbf{L}, \mathbf{R}]$ implies simultaneous computation of Part and Temporal Affinity Fields through a single CNN. For Model III, the channels corresponding to PAF are then passed for keypoint estimation along with VGG features from current frame and keypoints from previous frame. As Model III consists of only three blocks, it has the fastest inference, however it proved to be the most difficult to train.

3.2. Topology of Spatio-Temporal Affinity Fields

For our body model, we define $K = 21$ body parts or keypoints which is the union of body parts in COCO and MPII pose datasets. They include ears, nose and eyes from COCO; and head and neck from MPII. Next, there are several possible ways to associate and track the keypoints and

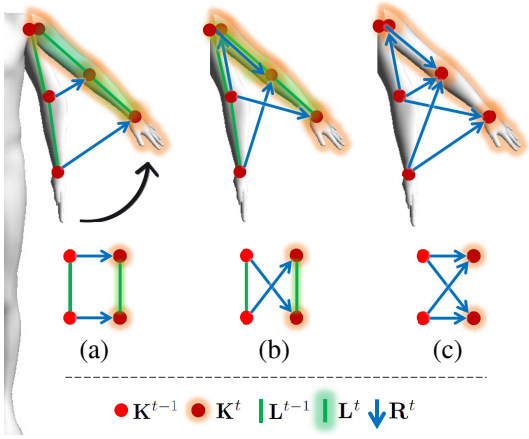

Figure 3: This figure illustrates the three possible topology variations for Spatio-Temporal Affinity Fields including the new cross-linked limb topology (b). The red and green color indicates previous frame and current frame, respectively. Keypoints, PAFs and TAFs are represented by solid circles, straight lines and arrows, respectively.

STAFs across frames as illustrated in Figure 3. In this figure, solid circles represent keypoints while straight lines and arrows stand for PAFs and TAFs, respectively. Figure 3(a) consists of TAFs between same keypoints as well as PAFs. For this topology, the number of TAFs and PAFs is 21 and 48, respectively. The TAFs capture temporal connections directly across keypoints similar to [7].

On the other hand, Figure 3(b) consists of TAFs between different limbs in a cross-linked manner across frames. The number of PAFs and TAFs is 48 and 96, respectively. We also tested the topology in Figure 3(c) which consists of 69 keypoints and limb TAFs only. This does not model any spatial links within frames across keypoints.

3.3. Model Training

During training, we unroll each model to handle multiple frames at once. Each model is first pre-trained in **Image Mode** where we present a single image or frame at each time instant to the model. This implies multiple applications of PAF and keypoint stages to the same frame. We train with COCO, MPII and PoseTrack datasets with a batch distribution of 0.7, 0.2 and 0.1, respectively, corresponding dataset sizes where each batch consists of images or frames from one dataset exclusively. For masking out un-annotated keypoints, we use the head bounding boxes available in MPII and Posetrack datasets, and location of annotated keypoints for batches from COCO dataset. The net takes in 368×368 images and has scaling, rotation and translation augmentations. Heatmaps are computed with an ℓ_2 loss with a stride of 8 resulting in 46×46 dimensional heatmaps. In topology 3(b), we initialize the TAF with PAF, and zeros for 3(a). We train the net for a maximum of 400k iterations.

Next, we proceed training in the **Video Mode** where we expose the network to video sequences. For static image datasets including COCO and MPII, we augment data with video sequences that have length equal to number of times the network is unrolled by synthesizing motion with scaling, rotation and translation. We train COCO, MPII and PoseTrack in Video Mode with a batch distribution of of 0.4, 0.1 and 0.5, respectively. Moreover, we also use skip-frame augmentation for video-based PoseTrack dataset where some of the randomly selected sequences skip up to 3 frames. We lock the weights of VGG module in Video Mode. For Model I, we only trained the TAFs block when training on videos. For Model II, we trained keypoints, PAFs and TAFs for 5000 epochs, then locked all modules except TAFs. In Model III, both STAFs and keypoints remained unlocked throughout 300k iterations.

3.4. Inference and Tracking

The method described till now predicts heatmaps of keypoints and STAFs at every frame. Next, we present the framework to perform pose inference and tracking across frames given the predicted heatmaps. Let the inferred poses at time t be given by $\{\mathbf{P}^{t,1}, \mathbf{P}^{t,2}, \ldots, \mathbf{P}^{t,N}\}$ where the second superscript indexes over people in each frame. Each pose at a particular time consists of up to K keypoints that become part of a pose post inference, i.e., $\mathbf{P}^{t,n} = \{\overline{\mathbf{K}}_1^{t,n}, \overline{\mathbf{K}}_2^{t,n}, \ldots, \overline{\mathbf{K}}_K^{t,n}\}$.

The detection and tracking procedure begins with localization of keypoints at time t. The inferred keypoints $\overline{\mathbf{K}}^t$ are obtained by rescaling the heatmaps to original image resolution followed by non-maximal suppression. Then, we infer PAF, $\overline{\mathbf{L}}^t$, and TAF, $\overline{\mathbf{R}}^t$, weights between all pairs of keypoints in each frame defined by the given topology, i.e.,

$$\overline{\mathbf{L}}^t_{k\to k'}\omega(\overline{\mathbf{K}}^t_k, \overline{\mathbf{K}}^t_{k'}), \quad \overline{\mathbf{R}}^t_{k\to k'}\omega(\overline{\mathbf{K}}^{t-1}_k, \overline{\mathbf{K}}^t_{k'}), \qquad (5)$$

where the function $\omega(\cdot)$ samples points between the two keypoints, computes the dot product between the the mean vector of the sampled points and the directional vector from the first to the second keypoint.

Both the inferred PAF and TAF weights are sorted by their scores before inferring the complete poses and associating them across frames with unique ids. We perform this in a bottom-up style where we utilize poses and inferred PAFs from the previous frame to determine the update, addition or deletion of tracklets. Going through each PAF in the sorted list, (i) we initialize a new pose if both keypoints in the PAF are unassigned, (ii) add to existing pose if one of the keypoints is assigned, (iii) update score of PAF in pose if both are assigned to the same pose, and (iv) merge two poses if keypoints belong to different poses with opposing keypoints unassigned. Finally, we assign id to each pose in the current frame with the most frequent id of keypoints from the previous frame. For cases where we have

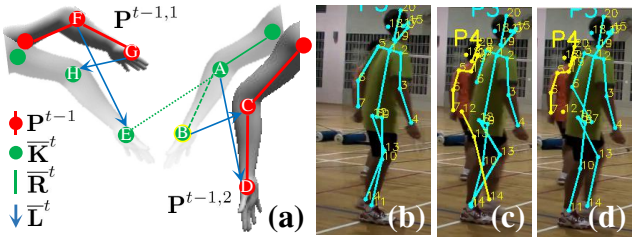

Figure 4: (a) Ambiguity when selecting between two wrist locations B and E is resolved by reweighing PAFs through TAFs. (b)-(d): With transitivity, incorrect PAFs containing ankles (c) are resolved with past pose (b) resulting in (d).

ambiguous PAFs i.e., multiple equally likely possibilities as seen in Figure 4, we use transitivity that reweighs PAFs with TAFs to disambiguate between them, using α as a biasing weight. In this figure, keypoint $\{A\}$ - an elbow - is under consideration with wrists $\{B\}$ and $\{E\}$ as two possibilities. We select the strongest TAFs where $\{A, B, C, D, A\}$ has a higher weight than $\{A, E, F, G, A\}$, computed as:

$$\overline{\mathbf{L}}^{t,n}_{k\to k'} = (1-\alpha) * \omega(\overline{\mathbf{K}}^{t-1,n}_k, \overline{\mathbf{K}}^{t,n}_{k'}) + \alpha * \omega(\overline{\mathbf{K}}^{t,n}_k, \overline{\mathbf{K}}^{t,n}_{k'}).$$

4. Experiments

In this section, we present results of our experiments. Input images to networks are resized at Wx368 maintaining aspect ratio for single scale (SS); and Wx736, Wx368 and Wx184 for multiple scales (MS). The heatmaps for multiple scales are re-sized back to Wx736 and merged through averaging. This is followed by inference and tracking.

4.1. Ablation Study

We conducted a series of ablation studies to determine the construction of our network architecture:

Filter Sizes: We studied the effect of filter size in each of the modules. As discussed in Sec. 3, each block either consists of five 7×7 layers followed by two 1×1 layers [5], or each 7×7 layer is replaced with three 3×3 layers in the alternate experiment. The results are shown in Table 7. We run single frame inference on Model I and find the 3×3 filter size to be 2% more accurate than 7×7, with significant boosts in average precision of knee and ankle keypoints. It is also 40% faster while requiring 40% more memory.

Method	Hea	Sho	Elb	Wri	Hip	Kne	Ank	mAP	fps
Model I - 3x3	75.7	73.9	67.8	56.3	66.8	62.3	56.9	66.3	14
Model I - 7x7	76.0	73.3	66.4	54.0	63.4	59.2	52.2	64.3	10

Table 1: This table shows results for experiments with the two filter sizes on PoseTrack 2017 validation set.

Video Mode / Depth of First Stage: Next, we report results when training in *Image Mode (Im)* using single images, and

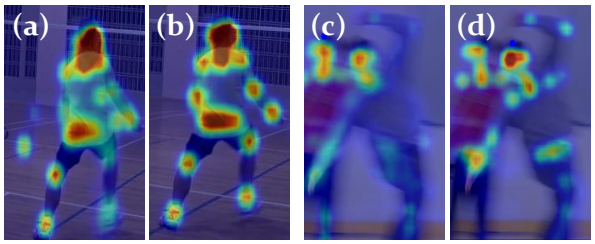

Figure 5: Improvement in quality of heatmaps before (a,c) and after (b,d) the network is exposed to videos and synthetic motion augmentation. We observe better peaks and less noise across both PAF and keypoint heatmaps.

when we continue training beyond images while exposing the network to videos and augmentation with synthetic motion in the *Video Mode (Vid)*. During testing, the network is run recurrently on video sequences with one frame per stage. Model II is deployed for these experiments. We find that by exposing the network to video sequences for 5000 iterations, we were able to boost the mAP as seen in Table 2 and Fig. 9. We also find that if we use the same depth, i.e., number of channels for the first frame as the other frames (128-128), the network was not able to generalize well to recurrent execution (56.6 mAP) when trained with Image Mode. When reducing the depth for the first frame to one-half, i.e. (64-128), we found that the generalization to videos was better (62.6 mAP). When trained with Video Mode, mAP increased further to 64.1. We reason that the 64-depth modules produced relatively vague outputs which gave sufficient room for the subsequent modules in the following frames to process and refine the heatmaps yielding a boost in performance. Furthermore, this also highlights the importance of incorporating shot change detection and running first stage at each shot change.

Method	Hea	Sho	Elb	Wri	Hip	Kne	Ank	mAP	fps
Im - 7x7 - 128-128	74.6	69.6	55.5	40.2	56.4	47.2	44.0	56.6	27
Vid - 7x7 - 128-128	76.2	71.6	64.5	51.9	62.6	59.3	52.5	63.6	27
Im - 7x7 - 64-128	73.5	72.2	63.8	52.1	62.7	57.3	51.1	62.6	27
Vid - 7x7 - 64-128	75.8	73.4	65.5	53.8	64.2	58.4	51.4	64.1	27
Im - 3x3 - 64-128	73.5	72.5	65.0	52.7	63.7	57.7	53.2	63.4	35
Vid - 3x3 - 64-128	75.4	73.2	67.4	55.0	63.9	58.4	53.5	64.6	35

Table 2: This table shows single-scale performance using Model II before and after training with videos, filter sizes, as well as different depths for first stage.

Effect of Camera Frame Rate on mAP: For these experiments, we studied how the frame rate of the camera and number of stages affect the accuracy of pose estimation. With a high frame rate, the apparent motion between frames is smooth, while relatively abrupt with a low frame-rate camera. Therefore, the heatmaps from previous frames would not be as useful at low frame rates. We tested this hypothesis considering Model I (five stages on the same

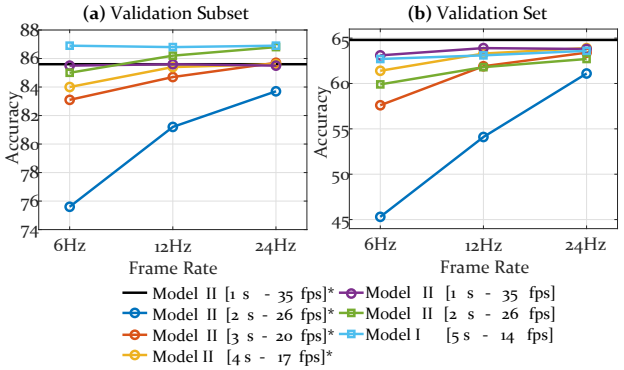

Figure 6: These graphs show mAP curves as a function of frame rates of camera, i.e., the rate at which an original 24Hz video is input to the method. The flat black line shows the performance of five-stage Model I, while '*' in the legend indicates training using Image Mode only.

modules without ingesting previous heatmaps), and Model II (different number of stages with each ingesting heatmaps from previous frame). We also evaluate the influence of training with Image and Video modes in Figure 6.

Fig. 6(a) shows results on a subset of ten sequences where the human subjects comprised at least 30% of the frame height in the PoseTrack 2017 validation set. Fig. 6(b) presents results on the entire validation set. The original videos were assumed to run at the film-standard 24 Hz hence we ran experiments by varying frame rates at 24, 12 and 6 Hz through sub-sampling. The ground truth has been annotated at 6 Hz. As expected, accuracy is proportional to video frame rate and number of stages. When the Model II was trained in Image Mode, we observed small increments in accuracy until at four stages, it peaks at the same level as Model I. Upon training with Video Mode, it surpasses this accuracy peaking earlier at two stages.

When considering the entire validation set, the approach is still able to reap the benefits of more stages and training in Video Mode as can be seen in Fig. 6(b). However, it was barely able to reach the accuracy of the much slower Model I. For the validation set, the accuracy was harmed when including sequences with smaller apparent size of humans. These sequences usually were more crowded as well and passing in the previous heatmaps seemed to hurt the performance. The body parts of small-sized humans only occupied a few pixels in the heatmap and the normalized direction vectors were inconsistent and random across frames.

Influence of Topology Type in Tracking: Next, we report the ablation experiments on tracking performance evaluated using Multiple Object Tracking Accuracy (MOTA) metric in Table 3. We evaluate results using Topology A and B from Fig. 3(a) and 3(b), respectively, both with Models I and II and found an improvement in tracking using limb

TAF in Topology B versus keypoints TAF in Topology A. As highlighted in Fig. 1, Topology A does not have associative properties when a keypoint has minimal motion or when a new person appears. Although we enforced proximity assumption across time t and $t-1$ that keypoints less than 2 pixels apart should be associated and adjusted it according to scale (similar to [7]), however, this still resulted in false positives since it is difficult to disambiguate between a newly detected person and some nearby stationary person. Furthermore, where the motion of a person tended to be small, Topology A resulted in very jittery and noisy vectors causing more reliance on pixel distances. This was further exacerbated by recurrence where accumulation of noisy vectors from previous frame heatmaps deteriorated associative ability of Temporal Affinity Fields.

Topology B solves all of these problems elegantly. The longer cross-linked limb TAF connections preserve information even in the absence of motion or appearance of new people since the TAF effectively collapses to a PAF in such cases. This allows us to avoid association heuristics and makes the problem of new person identification trivial. With this representation, recurrence was observably beneficial due to true and consistent representation irrespective of magnitude of motion. As a side-advantage, this also allowed us to warm-start the TAF input with PAF providing more reliable initialization for tracking in the first frame.

For Model III, training beyond 5000 iterations gradually begins to harm the accuracy of the pose estimation resulting in reduced tracking performance as well. This is primarily due to the disparity in the amount of diverse data between COCO/MPII and Posetrack datasets. For Model II, if we train on keypoints and PAFs modules and lock their weights afterwards, then follow with training only the TAF, this results in better performance with a significant boost in speed as well. However, Model I outperformed the other models with five stages for keypoints and PAFs; and a single recurrent stage for TAF. However this comes at the expense of speed. Furthermore, we observe that an increase in mAP ends up sub-linearly increasing the MOTA as well.

Method	Wrist-AP	Ankles-AP	mAP	MOTA	fps
Model I-A	56.2	56.4	66.0	58.5	14
Model I-B	56.3	56.9	66.3	59.4	13
Model II-A	54.9	53.0	64.4	57.4	28
Model II-B	55.0	53.5	64.6	58.4	27
Model III-B	51.9	49.5	61.6	57.8	30

Table 3: This table shows pose estimation and tracking performance for combinations of model types and topologies.

Effect of Video Rate and Number of People on Tracking: Finally, we performed a study on how the frame rate of the camera affects tracking accuracy, since a lower frame rate would require longer associations in pixel space.

We ran Lukas Kanade (LK) as a baseline tracker by replacing the TAF Module in Model I with LK (21×21 window and 3 pyramid levels). Initially, we observe that there is a roughly 2.0% improvement in MOTA as seen in Fig. 7(a). However, with careful observation we note that around 20% of the sequences have significant articulation and camera movement, where TAF outperformed LK as it was not able to match keypoints across large displacements whereas TAF found matches due to its stronger descriptive power. TAF was able to maintain tracking accuracy even with low frame-rate cameras, but with LK the MOTA drops off significantly (see Fig. 7(a)). Furthermore, Fig. 7(b) suggests that our approach is nearly run-time invariant to number of people in the frame making it suitable for crowded scenes.

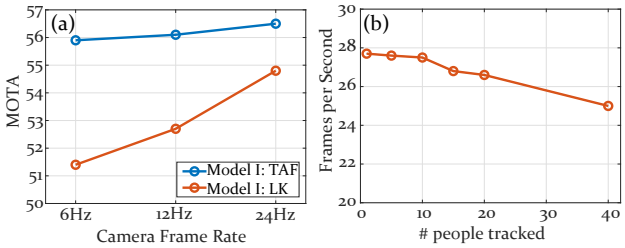

Figure 7: (a) This graph shows MOTA as a function of video frame rate for Temporal Affinity Fields (TAF) and Lukas-Kanade (LK) tracker. The performance of TAF is virtually invariant to frame rate or alternatively to the amount of motion between frames. (b) Our approach is effectively run-time invariant to the number of people in the scene.

4.2. Comparison

We present results on PoseTrack dataset in Table 4 for 2017 validation set (top), 2017 test set (middle) and 2018 validation set (bottom). FlowTrack, JointFlow and PoseFlow are included as comparison in this table. FlowTrack is a top-down approach which means human detection is performed first followed by pose estimation. Due to this reason, it is significantly slower than bottom-up approaches such as ours. Model II-B with single scale is competitive with other bottom-up approaches while being 270% faster. However, multi-scale (MS) processing boosts performance by $\sim$6% and $\sim$1.5% for mAP and MOTA, respectively. We are also able to achieve competitive results on the PoseTrack 2018 Validation set while maintaining the best speeds amongst all reported results. Note that PoseTrack 2018 Test set was not released to public at the time of submission of this paper. Figure 8 shows some qualitative results.

5. Conclusion

In this paper, we first outlined why the recurrent Spatio-Temporal Affinity Fields (STAFs) is the right approach for detection and tracking of articulated human pose in videos,

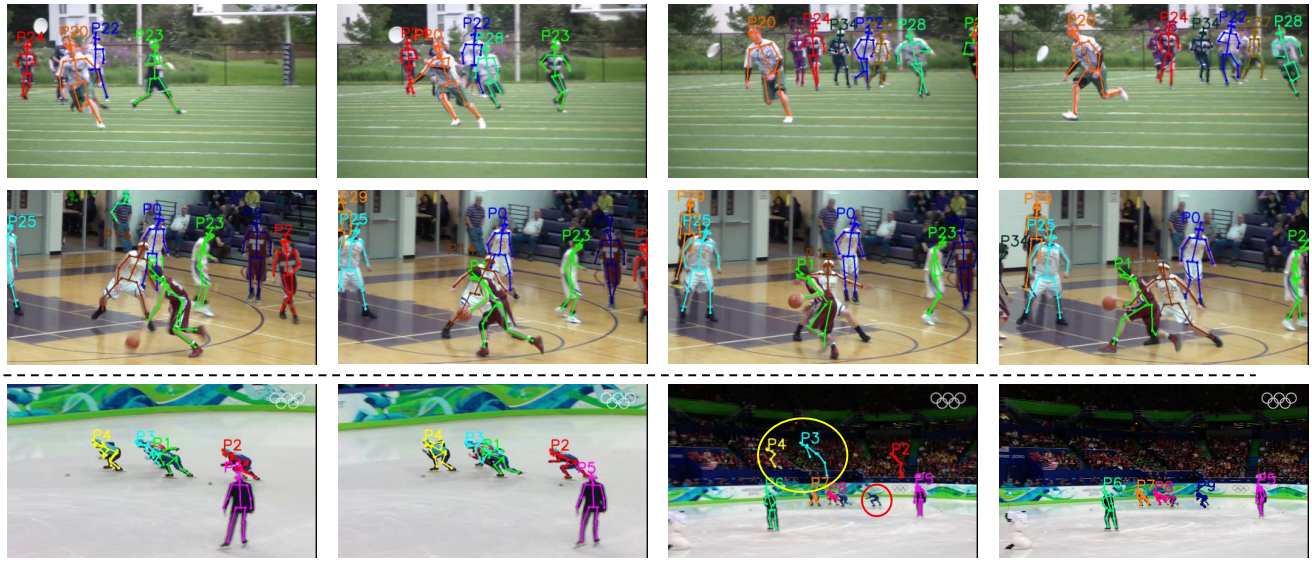

Figure 8: Three example cases of tracking at ∼30 FPS on multiple targets. **Top/Middle:** Observe that tracking continues to function despite large motion displacements and occlusions. **Bottom:** A failure case where abrupt scene change causes ghosting, where previous scene person appears in new frame. Note that small targets also take several stages to fully appear, hence a warm start will be required during shot change.

	Method	Wrist-AP	Ankles-AP	mAP	MOTA	fps
	Posetrack 2017 Validation					
Top-Down	Detect-and-track [8]	51.7	49.8	60.6	55.2	1.2
Top-Down	FlowTrack - 152 [37]	72.4	67.1	76.7	65.4	-
Top-Down	FlowTrack - 50 [37]	66.0	61.7	72.4	62.9	-
Top-Down	MDPN - 152 [11]	77.5	71.4	80.7	66.0	-
Bottom-Up	PoseFlow [38]	61.1	61.3	66.5	58.3	10*
Bottom-Up	JointFlow [7]	-	-	69.3	59.8	0.2
Bottom-Up	Model II-B (SS)	55.0	53.5	64.6	58.4	27
Bottom-Up	Model I-B (SS)	56.8	56.8	66.3	59.4	13
Bottom-Up	**Model II-B (MS)**	**62.9**	**60.9**	**71.5**	**61.3**	**7**
Bottom-Up	**Model I-B (MS)**	**65.0**	**62.7**	**72.6**	**62.7**	**2**
	Posetrack 2017 Testing					
Top-Down	Detect-and-track [8]	-	-	59.6	51.8	1.2
Top-Down	Flowtrack - 152 [37]	70.7	64.9	73.9	57.6	-
Top-Down	Flowtrack - 50 [37]	65.1	60.3	70.0	56.4	-
Bottom-Up	PoseTrack [2]	54.3	49.2	59.4	48.4	-
Bottom-Up	BUTD [18]	52.9	42.6	59.1	50.6	-
Bottom-Up	PoseFlow [38]	59.0	57.9	63.0	51.0	10*
Bottom-Up	JointFlow [7]	53.1	50.4	63.3	53.1	0.2
Bottom-Up	**Model II-B (MS)**	**62.8**	**59.5**	**69.6**	**52.4**	**7**
Bottom-Up	**Model I-B (MS)**	**65.0**	**60.7**	**70.3**	**53.8**	**2**
	Posetrack 2018 Validation					
Bottom-Up	Model II-B (SS)	56.2	54.2	63.7	58.4	27
Bottom-Up	Model I-B (SS)	58.3	56.7	64.9	59.6	13
Bottom-Up	Model II-B (MS)	62.7	60.6	69.9	59.8	7
Bottom-Up	Model I-B (MS)	64.7	62.0	70.4	60.9	3

Table 4: This table shows comparison on the Posetrack dataset. For our approach, we report results with Models I / II and Topology B. 'SS' and 'MS' refer to single and multiple scales, respectively. The last column shows the speed in frames per second (* excludes pose inference time). FlowTrack is a **top-down** approach that uses ResNet with 152 and 50 layers; whereas JointFlow, PoseFlow and our approach are **bottom-up**.

especially for real-time reactive systems. We demonstrated this by showing that leveraging the previous frame data within a recurrent structure and training on video sequences yields equally good results as a multi-stage network albeit at much lower computation cost. We also demonstrated the stability of tracking accuracy at reduced frame rates for the TAF formulation due to its ability to correlate keypoints over large pixel distances. This implies that our method can be deployed on low-power embedded systems that may not be able to run large networks at high frame rates, yet are able to maintain reasonable accuracy. We also demonstrated our new cross-linked limb temporal topology is able to generalize better than previous approaches due to strong associative power with PAF being a special case of TAF. We are also able to operate at the same consistent speed irrespective of the number of people due to bottom-up formulation. Our method is currently the most efficient and the best bottom-up fully *online* detection and tracking approach for articulated human poses. For future work, we plan to embed a re-identification module to handle cases of people leaving the camera view for long duration of time and reappearing later in time. Furthermore, detecting shot change and triggering warm-starting at every shot change has the potential to boost pose estimation and tracking performance.

Notes: More details on algorithms are in the supplementary document. SMPL [22] used for figures. The runtime code for this will be released into OpenPose [13] in the near future.

(Supplementary Material)

Yaadhav Raaj Haroon Idrees Gines Hidalgo Yaser Sheikh
The Robotics Institute, Carnegie Mellon University
{raaj@cmu.edu, hidrees@andrew.cmu.edu, gines@cmu.edu, yaser@cs.cmu.edu}

6. Video

A Video has been made to demonstrate this paper. A low res version has been included due to space constraints, or a high res version can be found in the following: Video Link. All Sequences observed in this video were generated with Model III at single-scale in real time at 30 FPS.

7. Model Training

During training, we unroll each model so that it can handle multiple frames at once. Each model is first pre-trained in **Image Mode** where we present a single image or frame at each time instant to the model. This implies multiple applications of PAF/KP stages to the same frame. We train with COCO, MPII and PoseTrack datasets with a batch distribution of 0.7, 0.2 and 0.1, respectively, matching dataset sizes, where each batch consists of images or frames from one dataset exclusively. We use the head bounding box information in MPII and Posetrack datasets to mask out the eyes/nose/ears in the background heatmap channel when considering the MPII and PoseTrack batches, and mask out the neck and top head positions using the annotated eyes/nose/ears keypoints for COCO batches. The net is input images of 368×368 dimensions and has scaling, rotation and translation augmentations, where regions not annotated are masked out. Heatmaps are computed with an ℓ_2 loss with a stride of 8 resulting in 46×46 dimensional heatmaps. In topology (b) and (c), we initialize the TAF with PAF, and zeros for (a). We train the net for max of 400k iterations.

Next, we proceed training in the **Video Mode** where we expose the network to video sequences. For static image datasets including COCO and MPII, we augment data with video motion sequences by synthesizing motion with scaling, rotation and translation over the unroll count. We train COCO, MPII and PoseTrack in Video Mode with a batch distribution of of 0.4, 0.1 and 0.5, respectively. Moreover, we also use skip-frame augmentation for video-based PoseTrack dataset, where some of the randomly selected sequences skip up to 3 frames. We lock the weights of VGG module in Video Mode and only train the STAFs and keypoints blocks. For Model I, we only trained the TAFs block when training on videos. For Model II, we trained PAFs, keypoints and TAFs for 5000 epochs, then locked PAFs and keypoints before training TAFs only. In Model III, both STAFs and keypoints were kept unlocked and were trained for 300k iterations.

Param	Image Mode	Video Mode
Input Resolution	368x368	368x368
Heatmap Resolution	46x46	46x46
Data Dist. (COCO, MPII, PT)	0.7,0.2,0.1	0.4,0.1,0.5
Frame Skip Augmentation	-	3
Scaling	$[0.7x, 1.3x]$	$[0.7x, 1.3x]$
Rotation	$[-30°, 30°]$	$[-20°, 30°]$
Translation	$[-30, 30]$	$[-50, 50]$
VGG Learning Rate	0.00004	0.0
PAF/TAF/KP Learning Rate	0.00008	0.00004
Solver	Adam	Adam
Momentum and Decay β_1, β_2	0.9, 0.999	0.9, 0.999
Decay	0.0005	0.0005
Step	[150k, 250k, 360k]	[100k, 200k, 250k]
Step Size	0.5	0.5
Total Epochs	400k	300k

Table 5: Training Parameters for both Image Mode and Video Mode

7.1. Inference and Tracking

The method described till now predicts heatmaps of keypoints and STAFs at every frame by running CNNs associated with each module while passing required data computed from previous frame. Next, we present the framework to perform pose inference as well as tracking across frames given the output heatmaps. Let the inferred poses at time t and $t-1$ be given by:

$$\begin{aligned} \mathbf{P}^{t} &= \{\mathbf{P}^{t,1}, \mathbf{P}^{t,2}, \ldots, \mathbf{P}^{t,N}\}, \\ \mathbf{P}^{t-1} &= \{\mathbf{P}^{t-1,1}, \mathbf{P}^{t-1,2}, \ldots, \mathbf{P}^{t-1,M}\}, \end{aligned} \tag{6}$$

where the second superscript indexes over people in each frame. Each pose at a particular time $\mathbf{P}^{t,n}$ consists of up to K keypoints post inference, i.e., $\mathbf{P}^{t,n}$ includes only those keypoints that become part of a pose:

$$\mathbf{P}^{t,n} = \{\overline{\mathbf{K}}_1^{t,n}, \overline{\mathbf{K}}_2^{t,n}, \ldots, \overline{\mathbf{K}}_K^{t,n}\}. \tag{7}$$

The detection and tracking procedure begins with localization of keypoints at time t. The inferred keypoints $\overline{\mathbf{K}}^t$ are obtained by rescaling the heatmaps to match the original image resolution followed by non-maximal suppression. Then, we infer PAF and TAF weights between all possible pairs of keypoints in each frame defined by the given topology, i.e.,

$$\overline{\mathbf{L}}^t_{k\to k'}\omega(\overline{\mathbf{K}}^t_k, \overline{\mathbf{K}}^t_{k'}), \quad \overline{\mathbf{R}}^t_{k\to k'} \tag{8}$$

where the function $\omega(\cdot)$ samples points between the two argument keypoints, computes the dot product between directional vector among the two points and the mean vector of the sampled points. The inference of PAF, $\overline{\mathbf{L}}^t$, and TAF, $\overline{\mathbf{R}}^t$, weights is constrained by the spatio-temporal topology, where the spatial and temporal constraints are encoded in tables $\ddot{\mathbf{L}}$ and $\ddot{\mathbf{R}}$, respectively.

$$max\left(\sum_t^N\sum_k^K\left(\overline{\mathbf{L}}^{t,n}_{k\to k'} + \overline{\mathbf{R}}^t_{k\to k'}\right)\right) \tag{9}$$

Overall, we wish to generate a set of people $\mathbf{P}^t$ with ids that maximizes the connection scores between their keypoints pairs $(\overline{\mathbf{K}}^{t-1,n}_k, \overline{\mathbf{K}}^{t,n}_{k'})$, and temporal connections given previous keypoints $\overline{\mathbf{K}}^{t-1,n}$ from tracklets $\mathbf{P}^{t-1,n}$ given an association.

To do this, both the inferred PAF and TAF weights are computed then sorted by their scores before inferring the complete poses, $\mathbf{P}^t$, and associating them across frames with unique ids. We perform this in a bottom-up style as described in Algorithm 1 where we utilize $\overline{\mathbf{R}}^t$ and $\mathbf{P}^{t-1}$ to determine the update, addition or deletion of tracklets. Going through each PAF in the sorted list, (i) we initialize a new pose if both keypoints in the PAF are unassigned, (ii) add to existing pose if one of the keypoints is assigned, (iii) update score of PAF in pose if both are assigned to the same pose, and (iv) merge two poses if keypoints belong to different poses with opposing keypoints unassigned. Finally, we assign id to each pose in the current frame with the most frequent id of keypoints from the previous frame. This is done over all tracklets and people very quickly as it is done on the GPU.

Furthermore, we make use of past poses $\mathbf{P}^{t-1}$ and TAFs $\overline{\mathbf{R}}^t$ to reweigh PAFs. For cases where we have an ambiguous PAFs (Alg. 1, 7:) as seen in Figure 4, we use transitivity that reweighs PAFs to disambiguate between them. In this figure, keypoint $\{A\}$ - an elbow - is under consideration, with wrists $\{B\}$/$\{E\}$ as two possibilities. We select the strongest TAFs where $\{A, B, C, D, A\}$ has a higher weight than $\{A, E, F, G, A\}$.

$$\overline{\mathbf{L}}^{t,n}_{k\to k'} = (1-\alpha)\omega(\overline{\mathbf{K}}^{t-1,n}_k, \overline{\mathbf{K}}^{t,n}_{k'}) + \alpha * \omega(\overline{\mathbf{K}}^{t,n}_k, \overline{\mathbf{K}}^{t,n}_{k'}). \tag{10}$$

Algorithm 1 : Estimation and tracking of keypoints and STAFs

Input: $\mathbf{K}^t, \mathbf{L}^t, \mathbf{R}^t$, and $\mathbf{P}^{t-1}$ with unique ids
Output: $\mathbf{P}^t$ with ids

1: **procedure** INFERPOSES()
2: Compute $\overline{\mathbf{L}}^t$ given $\overline{\mathbf{K}}^t$, $\mathbf{L}^t$ and $\ddot{\mathbf{L}}$
3: Compute $\overline{\mathbf{R}}^t$ given $\overline{\mathbf{K}}^t$, $\mathbf{R}^t$, $\mathbf{P}^{t-1}$ and $\ddot{\mathbf{R}}$
4: Sort $\overline{\mathbf{L}}^t$ and $\overline{\mathbf{R}}^t$ by score
5: Initialize empty map of people $\mathbf{P}^t$
6: **for** every $\overline{\mathbf{L}}^{t,n}_{k\to k'}$ in $\overline{\mathbf{L}}^t$ **do**
7: If k and k' unassigned; add new $\mathbf{P}^{t,n}$
8: If k or k' assigned; add to existing $\mathbf{P}^{t,n}$
9: If k and k' assigned; update score of $\mathbf{P}^{t,n}$
10: If k assigned to $\mathbf{P}^{t,n}$ and k' assigned to $\mathbf{P}^{t,n'}$
11: & $\mathbf{P}^{t,n}$ and $\mathbf{P}^{t,n'}$ lack opposing points;
12: merge $\mathbf{P}^{t,n}$ and $\mathbf{P}^{t,n'}$.
13: **end for**
14: **for** $\mathbf{P}^{t,n}$ in $\mathbf{P}^t$ **do**
15: **for** $\overline{\mathbf{K}}^{t,n}$ in $\mathbf{P}^{t,n}$ **do**
16: Find $\overline{\mathbf{R}}^t$ with the highest score; copy id
17: **end for**
18: Update $\mathbf{P}^{t,n}$ with the most frequent id
19: **end for**
20: If insufficient keypoint matches for $\mathbf{P}^{t,n}$;
21: initialize tracklet
22: Remove $\mathbf{P}^{t-1,n}$ if no association made
23: **end procedure**

8. Additional Experiments

We present some experiments that were otherwise not displayed in detail in the main paper.

For the sake of completion, we first report results on the COCO dataset in Table 6. Despite using single set of weights for all the stages, we were able to get close results. Our network is designed to be lightweight and work in a recurrent fashion, so our main reference point is still the Posetrack datasets.

Filter Sizes: We observed that having each 7×7 filter replaced with a three 3×3 filter resulted in better accuracies, especially for knees and ankles. The results are shown in Table 7. We run single frame inference on Model I and find the 3×3 to be 2% more accurate than 7×7, with significant boosts in average precision of knee and ankle keypoints.

Recurrence of Keypoints Module: To verify that the network was indeed benefiting from ingesting previous frame heatmaps, we explicitly train a model where the keypoint module was connected to current PAF module in an *auxiliary* fashion and did not receive previous heatmaps (first

Method	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}
Top-Down Approaches					
Megvii [6]	73.0	91.7	80.9	69.5	78.1
G-RMI [27]	71.0	87.9	77.7	69.0	75.2
Mask R-CNN [12]	69.2	90.4	76.0	64.9	76.3
Bottom-Up Approaches					
PersonLab [26]	68.7	89.0	75.4	64.1	75.5
Associative Emb. [24]	65.5	86.8	72.3	60.6	72.6
OpenPose 2018 [13]	64.4	86.5	70.2	61.5	68.8
Proposed	61.5	82.2	67.1	58.5	66.7

Table 6: Results on the COCO test-dev dataset. Top: top-down results. Bottom: bottom-up results (top methods only). AP^{50} is for OKS $= 0.5$, AP^{L} is for large scale persons.

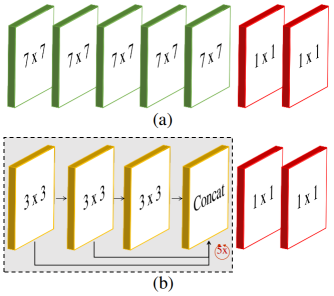

Figure 9: Types of filters usued

Method	Hea	Sho	Elb	Wri	Hip	Kne	Ank	mAP	fps
Model I - 3x3	75.7	73.9	67.8	56.3	66.8	62.3	56.9	66.3	14
Model I - 7x7	76.0	73.3	66.4	54.0	63.4	59.2	52.2	64.3	10

Table 7: This table shows results for experiments with the two filter sizes on PoseTrack 2017 validation set.

row of Table 8). The second row shows the case where keypoint module was *connected* to ingest output heatmaps from previous frames. The result is 2.1% improvement at single scale on the PoseTrack 2017 validation set using Model II with 7×7 filter size.

Method	Hea	Sho	Elb	Wri	Hip	Kne	Ank	mAP	fps
KP Auxiliary	72.3	71.2	63.9	51.4	60.1	56.3	50.0	61.5	28
KP Connected	76.2	71.6	64.5	51.9	62.6	59.3	52.5	63.6	27

Table 8: Performance using Model II where the keypoint module does not take feedback from previous heatmaps (auxiliary), and when it does ingest previous heatmaps (connected).

STAF Topology: We experimented with Topology A, B and C. Topology B proved to be better than A due to it's ability to preserve information even during minimal motion, lending itself better to the recurrent structure of our network. It especially performed better during jittery camera motion, or during crowded scenes with several people. Topology C, which does not consist of any current frame spatial information, was difficult to train and resulted in an MAP that was about 8% lower. This was mainly because we had to construct a person during the first frame, or during a new person appearance, and simply propagate it using the TAF, which proved to be less reliable than extracting poses on every frame with PAF/TAF, then propagating it with TAF.

9. Implementation

Training: We train on 4 Titan XP in Caffe. Testing: We test on a single 1080 Ti, and i7 7800X. We write our own code in C++ and CUDA.

References

[1] Posetrack leaderboard. https://posetrack.net/leaderboard.php. 2

[2] M. Andriluka, U. Iqbal, E. Insafutdinov, L. Pishchulin, and A. Milan. Posetrack: A benchmark for human pose estimation and tracking. *CVPR*, 2018., 2018. 9

[3] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2d human pose estimation: new benchmark and state of the art analysis. In *CVPR*. 2014. 1, 2

[4] M. Andriluka, S. Roth, and B. Schiele. Monocular 3D pose estimation and tracking by detection. In *CVPR*, 2010. 2

[5] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017. 1, 2, 5

[6] Y. Chen, Z. Wang, Y. Peng, Z. Zhang, G. Yu, and J. Sun. Cascaded pyramid network for multi-person pose estimation. *arXiv preprint arXiv:1711.07319*, 2017. 2, 12

[7] A. Doering, U. Iqbal, and J. Gall. Joint flow: Temporal flow fields for multi person tracking. *CoRR*, abs/1805.04596, 2018. 2, 4, 7, 9

[8] R. Girdhar, G. Gkioxari, L. Torresani, M. Paluri, and D. Tran. Detect-and-track: Efficient pose estimation in videos. *CoRR*, abs/1712.09184, 2017. 2, 9

[9] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He. Detectron. https://github.com/facebookresearch/detectron, 2018. 2

[10] G. Gkioxari, B. Hariharan, R. Girshick, and J. Malik. Using k-poselets for detecting people and localizing their keypoints. In *CVPR*, 2014. 2

[11] H. Guo, T. Tang, G. Luo, and Y. L. Riwei Chen. Multi-domain pose network for multi-person pose estimation and tracking. *ECCV*, 2018., 2018. 9

[12] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017. 2, 12

[13] G. Hidalgo, Z. Cao, T. Simon, S.-E. Wei, H. Joo, and Y. Sheikh. OpenPose library. https://github.com/CMU-Perceptual-Computing-Lab/openpose. 8, 12

[14] E. Insafutdinov, M. Andriluka, L. Pishchulin, S. Tang, E. Levinkov, B. Andres, and B. Schiele. Arttrack: Articulated multi-person tracking in the wild. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1293–1301, 2017. 2

[15] U. Iqbal and J. Gall. Multi-person pose estimation with local joint-to-person associations. In *ECCV Workshops, Crowd Understanding*, 2016. 2

[16] U. Iqbal, A. Milan, and J. Gall. Posetrack: Joint multi-person pose estimation and tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2

[17] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *International Conf. on Computer Vision (ICCV)*, pages 3192–3199, Dec. 2013. 2

[18] S. Jin, X. Ma, Z. Han, Y. Wu, and W. Yang. Towards multi-person pose tracking: Bottom-up and top-down methods. *ICCV*, 2017., 2017. 9

[19] S. Johnson and M. Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *BMVC*, 2010. 2

[20] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 1

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollr, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, Zrich, 2014. Oral. 2

[22] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, Oct. 2015. 8

[23] Y. Luo, J. Ren, Z. Wang, W. Sun, J. Pan, J. P. Jianbo Liu, and L. Lin. LSTM pose machines. In *CVPR*, 2018. 2

[24] A. Newell, Z. Huang, and J. Deng. Associative embedding: End-to-end learning for joint detection and grouping. *NIPS*, 2017., 2017. 12

[25] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. *CoRR*, abs/1603.06937, 2016. 2

[26] G. Papandreou, T. Zhu, L.-C. Chen, S. Gidaris, J. Tompson, and K. Murphy. Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model. *arXiv preprint arXiv:1803.08225*, 2018. 2, 12

[27] G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Tompson, C. Bregler, and K. Murphy. Towards accurate multi-person pose estimation in the wild. In *CVPR*, 2017. 2, 12

[28] T. Pfister, J. Charles, and A. Zisserman. Flowing convnets for human pose estimation in videos. In *IEEE International Conference on Computer Vision*, 2015. 2

[29] L. Pishchulin, M. Andriluka, P. Gehler, and B. Schiele. Poselet conditioned pictorial structures. In *CVPR*, volume 1, 2013. 2

[30] L. Pishchulin, A. Jain, M. Andriluka, T. Thormahlen, and . B. Schiele. Articulated people detection and pose estimation: Reshaping the future. In *CVPR*, 2012. 2

[31] I. K. Riza Alp Guler, Natalia Neverova. Densepose: Dense human pose estimation in the wild. 2018. 2

[32] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3

[33] J. Song, L. Wang, L. Van Gool, and O. Hilliges. Thin-slicing network: A deep structured model for pose estimation in videos. In *CVPR*, 2017. 2

[34] M. Sun and S. Savarese. Articulated part-based model for joint object detection and pose estimation. In *ICCV*, 2011. 2

[35] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *CVPR*, 2016. 2

[36] M. Z. Weiyu Zhang and K. Derpanis. From actemes to action: A strongly-supervised representation for detailed action understanding. In *ICCV*, 2013. 2

[37] B. Xiao, H. Wu, and Y. Wei. Simple baselines for human pose estimation and tracking. *CoRR*, abs/1804.06208, 2018. 2, 9

[38] Y. Xiu, J. Li, H. Wang, Y. Fang, and C. Lu. Pose flow: Efficient online pose tracking. In *BMVC*, 2018. 2, 9

[39] Y. Yang and D. Ramanan. Articulated human detection with flexible mixtures of parts. In *TPAMI*, 2013. 2