COMOTION: CONCURRENT MULTI-PERSON 3D MOTION

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce an approach for tracking detailed 3D poses of multiple people from a single monocular camera stream. Our system maintains temporally coherent predictions in crowded scenes filled with difficult poses and occlusions. Rather than detect poses and associate them to current tracks, our model directly updates all tracked poses simultaneously given a new input image. We train on numerous single-image and video datasets with both 2D and 3D annotations to produce a model that matches the 3D pose estimation quality of state-of-the-art systems while performing faster and more accurate tracking on in-the-wild videos.

1 INTRODUCTION

Some of the most exciting applications of computing require understanding where people are and how they move. A single monocular camera can support a variety of human-centric applications if it is coupled with a system that can model the motion of all humans in its visual field. For the broadest range of applications, we are interested in systems that (a) see multiple people, (b) estimate their poses in 3D, (c) track their poses through time in a monocular video stream, even through occlusion, and (d) do so online, in a streaming fashion, processing each frame and updating everyone's estimated 3D poses without peeking into the future.

For such a system to work in the wild, it must handle cluttered scenes that are crowded with people. As people move about, the model must keep track of who is who from frame to frame in order to produce a coherent 3D motion estimate. This can be difficult as people occlude each other, pass behind objects, and step out of view. Maintaining accurate 3D pose estimates online is particularly challenging, as the model cannot leverage future context to fill in the gap for a missing observation. Instead, the model must forecast to the best of its ability and snap to the correct state as soon as the person is visible again.

Many existing approaches to pose tracking follow a two-stage detect-and-associate paradigm (Insafutdinov et al., 2017; Xiao et al., 2018; Doering et al., 2022; Rajasegaran et al., 2022). The idea is to run a state-of-the-art pose estimation system for each frame, and then link poses across frames using cues such as proximity and appearance. As with many two-stage approaches, the second-stage pose estimates and tracks can be undermined by flaws in the first stage. If a detection is missed in the first stage, some other mechanism must be introduced to update the pose. For example, one option is to fill in any missing poses offline using the context of the full track (Goel et al., 2023). However, this solution is not suitable for online applications that deal with live video.

We present CoMotion, a video-based approach that performs frame-to-frame pose updates in a fundamentally different manner. Following the *tracking by attention* paradigm (Meinhardt et al., 2022), we do not link independent per-frame detections – rather, we train a recurrent model that maintains a set of tracked 3D poses and updates them when a new frame arrives. To update the poses, CoMotion directly ingests the pixels of the new frame. There is no separate pose estimation module that is run first to provide candidate poses. The model uses whatever evidence is available in the pixels to update the poses of all people in the scene simultaneously – even those who may not be currently visible. One advantage of this approach is that CoMotion can learn to utilize subtle pixel cues. A pair of feet poking out may be insufficient to trigger an independent detection, but can still be quite informative if the model has been following that person's movement.

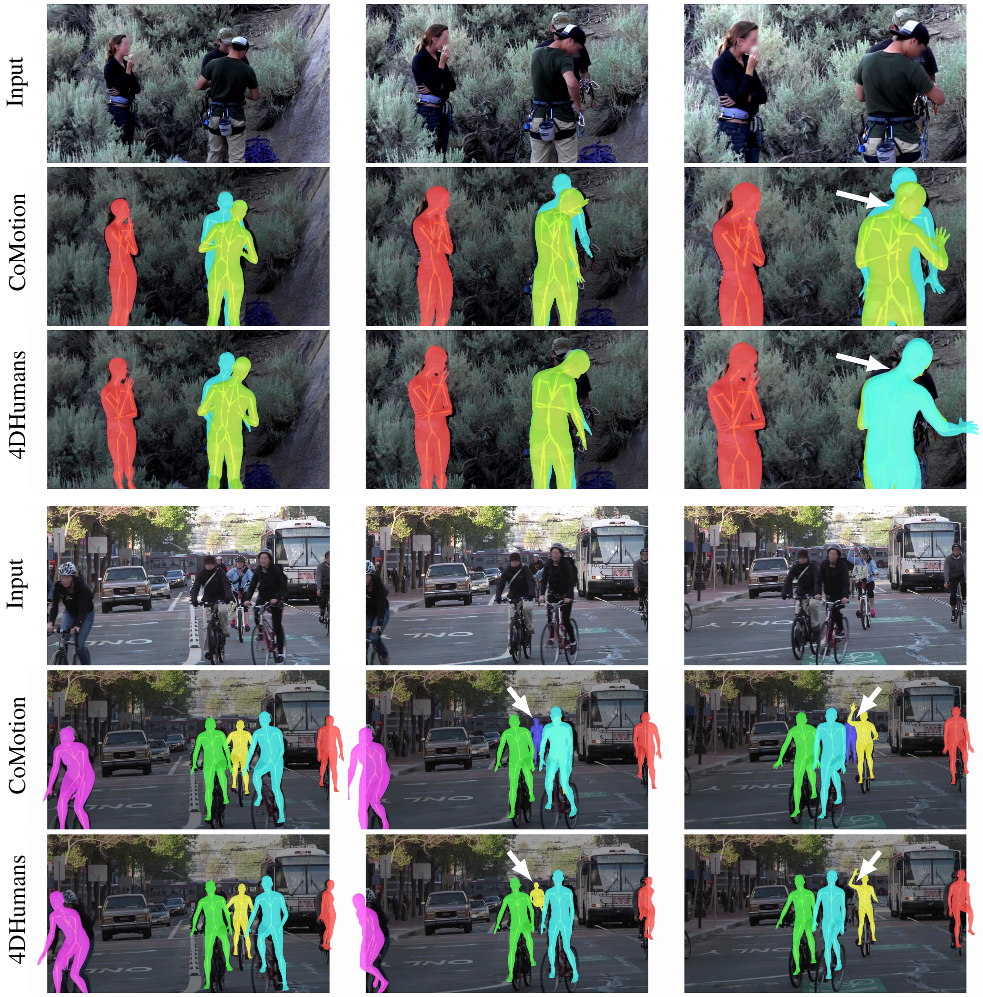

Figure 1: CoMotion tracks 3D poses online from monocular RGB video. Rather than detect new poses in each frame and associate them to existing tracks, CoMotion updates tracks directly from incoming image features. As a result, CoMotion keeps track of distinct individuals as they overlap in the camera frame (top) and occlude each other (bottom). Arrows highlight some points of interest.

Data is a challenge with such a video-based approach, for both training and evaluation. No dataset exists that provides ground-truth 3D poses of groups of people moving in diverse, cluttered real-world scenes. As a result, it is rare to find video-based 3D multi-person pose methods that support in-the-wild videos with large numbers of people. Comparable methods evaluate tracking on limited settings with a handful of people (Reddy et al., 2021; Sun et al., 2023). Only top-down methods report tracking results on more challenging real-world videos (Rajasegaran et al., 2022).

We find that we can achieve excellent performance by training on a heterogeneous mixture of datasets. Each dataset offers a complementary form of supervision – some are single-image, some are video, some are synthetic, and some provide partial ground truth. From 2D keypoint video datasets (Andriluka et al., 2018; Vendrow et al., 2023), our model learns robust tracking behavior, while from single-image pseudo-labeled data it learns accurate 3D pose estimation. We are unaware of any prior work in this area that trains on a comparably heterogenous mix of datasets. In particular, there are no multi-person 3D pose methods that supervise temporally unrolled predictions on complex videos such as those provided by PoseTrack (Andriluka et al., 2018; Doering et al., 2022).

We evaluate pose estimation performance and multi-person tracking using standard datasets for each task. CoMotion exhibits state-of-the-art pose estimation performance, while providing better and faster tracking with fewer temporal artifacts such as jitter and dropped poses. On Pose-

Track21(Doering et al., 2022), CoMotion improves MOTA by 8% and IDF1 by 10% relative to the prior state of the art. Figure 1 illustrates this combination of 3D pose estimation accuracy and tracking stability. At the same time, CoMotion is an order of magnitude faster than the prior state of the art.

2 Related Work

Many approaches to pose estimation and tracking have been explored. Some detect poses only in a single image, some operate on video but only handle a single person, and some track multiple people in video but don't estimate their poses. Few take on the holistic setting of predicting and tracking the poses of multiple people in video, let alone in a streaming (online) manner. For an accessible overview of the different perspectives, we classify relevant work along two dimensions, namely the input modality (single image or video) and the handling of multiple people.

Single-person, single-frame 3D pose. Many approaches to 3D human pose estimation predict a pose from a single image that is cropped to the target person (Bogo et al., 2016; Kanazawa et al., 2018; Pavlakos et al., 2018). Poses are commonly represented by the SMPL model (Loper et al., 2015), which defines a mesh from joint angles and shape parameters. The SMPL parameters can be regressed directly from an input image (Goel et al., 2023). Another common strategy is to employ a feedback loop that alternates between predicting SMPL parameters and observing the reprojection to image space to inform further updates (Zhang et al., 2021a; 2023a;b;c; Wang & Daniilidis, 2023). Much progress in this setting can be attributed to scaling up datasets, model sizes, and training times (Goel et al., 2023; Khirodkar et al., 2024).

Single-person pose from video. This setting assumes that a person has already been detected and tracked, such that it can be presented to the model through a sequence of cropped frames or bounding boxes. A line of work operates on pre-extracted 2D keypoints (Zheng et al., 2021; Zhang et al., 2022; Tang et al., 2023; Zhao et al., 2023) with transformers because the sparse input signal enables modeling of large temporal windows. Alternatively, recurrent networks enable online aggregation of features between frames. VIBE (Kocabas et al., 2020) and PMCE (You et al., 2023) use a GRU, and RoHM (Zhang et al., 2024) uses a denoising diffusion model to decode subsequent frames. Our approach also uses a GRU to run inference on video. However, where these prior methods rely on an external tracking solution and can only perform inference on a single subject, our method does its own tracking to handle multiple subjects in parallel.

Multi-person, single-frame pose. A straightforward approach to predicting poses of multiple people is detecting their individual bounding boxes and processing those via a single-person pose estimator (Goel et al., 2023; Fang et al., 2017; Papandreou et al., 2017; Huang et al., 2017). However, this approach scales poorly to crowded scenes. More recently, several works extract a dense feature map from the input image and perform cross-attention on a set of query tokens regressed from a ResNet or ViT applied to the same image (Liu et al., 2023; Shi et al., 2022; Baradel et al., 2024). In these approaches, the costly dense feature extractor is only run once per image. We also apply cross-attention, but our model produces queries in each timestep conditioned on past history and must also attend to people who may not be visible in the current frame (e.g., because they temporarily move out of the camera's field of view), which would not be relevant in the single-image setting.

Multi-person pose from video. A key challenge in multi-person pose from video is associating new observations with pre-existing tracks. An early approach was to estimate a graph of pose keypoints with intra-/inter-frame edge weights and then estimate connected components by solving a minimum-cost multi-cut problem (Insafutdinov et al., 2017). Better results were later obtained using a track-and-detect paradigm, where independent per-frame detections are grouped into tracks using visual keypoint features and image location (Girdhar et al., 2018) as well as 2D motion (Xiao et al., 2018). More recently, PHALP (Rajasegaran et al., 2022) adopted this mechanism for 3D pose rather than 2D, and also used the predicted 3D poses as an additional feature for grouping. 4D Humans (Goel et al., 2023) went further with a more general-purpose pose representation (Loper et al., 2015) and new architecture. These approaches achieve excellent results on modern benchmarks but scale poorly on videos with many subjects, since they run a pose estimator independently on each cropped detection. In contrast to these tracking-by-detection methods, CoMotion estimates poses and localizes existing tracks in parallel, reasoning over all poses simultaneously. In addition to a more holistic treatment of the scene, this approach is an order of magnitude faster.

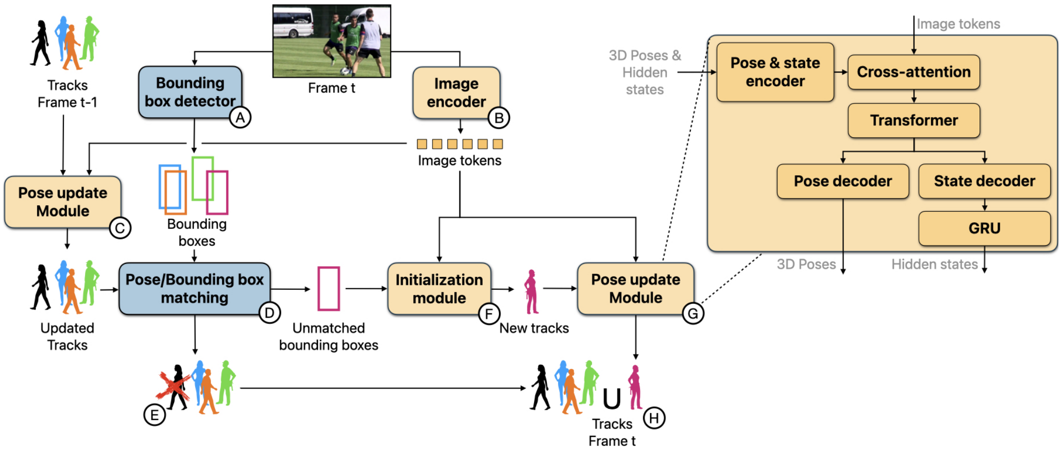

Figure 2: **Overview.** CoMotion estimates 3D poses for all people in a frame. Existing tracks from previous frames are updated Ⓒ and matched to detected bounding boxes Ⓓ produced by an off-the-shelf pretrained bounding box detector Ⓐ. Unmatched tracks are deleted Ⓔ and new tracks initialized Ⓕ from unmatched bounding boxes. An image encoder Ⓑ yields image tokens to instantiate new Ⓓ tracks for unmatched bounding boxes and update all tracks Ⓒ & Ⓖ. The heart of CoMotion is the pose update module Ⓒ & Ⓖ, which encodes tracks (3D poses and hidden states) to query image tokens and produce new poses and hidden states. The hidden state serves to provide learned features for the network to encode auxiliary information that is not included in the SMPL pose parameters. Fixed modules are blue, learnable modules are orange.

Unlike tracking-by-detection, tracking-by-attention (Meinhardt et al., 2022; Sun et al., 2020) detects new objects and updates existing tracks jointly by performing cross-attention between an image and a set of learned per-track and new-object query tokens. While many of these works condition the query tokens only on the previous few frames (Zeng et al., 2022), MeMOTR maintains a long-term memory of per-track query tokens and gradually updates them based on new observations (Gao & Wang, 2023). We follow the tracking-by-attention paradigm in this work, but find that there are many challenges to instantiating this paradigm for multi-person 3D pose tracking, including the nature of available training data. Bounding-box-based methods can get away with training on pairs of frames (Meinhardt et al., 2022), but we find that it is critical to unroll on longer video sequences during training to learn to track detailed 3D poses accurately.

3 CoMotion

Preliminaries. Given a sequence of monocular RGB images $\{I^1, I^2, ..., I^t\} \in \mathbb{R}^{h \times w \times 3}$, we aim to produce 3D pose estimates for each person in the scene at each time-step t conditioned only on the past and the present (no peeking into the future). Each prediction should be associated with a particular identity such that we know all poses belonging to a given individual. Predictions are provided as complete trajectories from the beginning to the end of a track. That is, there are no gaps in estimates due to temporary occlusions, although we do not support long-term reidentification.

We follow standard practice from prior work (Bogo et al., 2016; Kanazawa et al., 2018) and parameterize each person with SMPL (Loper et al., 2015), which consists of a translation term $\gamma \in \mathbb{R}^3$, joint angles $\theta \in \mathbb{R}^{72}$ (including a global orientation), and shape parameters $\beta \in \mathbb{R}^{10}$. As we focus our training and evaluation on 3D keypoint positions, we disregard other shape properties represented by β. That is, we expect our approach to accurately estimate bone lengths (which are determined by β), but not the actual body shape as this is not accurately reflected in the pseudo-ground truth SMPL annotations available for training.

All estimates are made in the camera coordinate frame. We do not explicitly model any changes to the camera pose or intrinsics (e.g., due to camera motion or zooming). During inference, the system accepts as input an intrinsics matrix K. All output 3D estimates will project correctly back into the image according to this provided matrix using the pinhole camera model. If no ground truth intrinsics are available, we fall back to reasonable defaults. To increase robustness against an incorrect intrinsics, we augment K during training. This allows us to make predictions on in-the-wild

Algorithm 1 CoMotion

```
tracks ← {}
for $I^t \leftarrow 1$ to $T$ do
    Ⓐ bboxes ← detector($I^t$)
    Ⓑ $F^t$ ← encoder($I^t$)
    if $len$(tracks) $> 0$ then                  ▷ Update tracks and compare to provided boxes
        Ⓒ tracks ← update($F^t$, tracks)
        Ⓓ bboxes$_{unmatched}$, tracks$_{unmatched}$ ← compareIOU(bboxes, tracks)
        Ⓔ tracks ← cleanup(tracks, tracks$_{unmatched}$)
        Ⓕ tracks$_{new}$ ← initialize($F^t$, bboxes$_{unmatched}$)
        Ⓖ tracks$_{new}$ ← update($F^t$, tracks$_{new}$)
        Ⓗ tracks ← concat(tracks, tracks$_{new}$)
    else                                         ▷ No existing tracks, start a new batch
        tracks ← initialize($F^t$, bboxes)
        tracks ← update($F^t$, tracks)
    end if
end for
```

videos by providing a generic setting for K while also using the correct intrinsics when they are available.

Overview. Figure 2 and Algorithm 1 provide an overview of the CoMotion architecture and logic. Given a new (RGB) frame t, an image encoder Ⓑ produces per-pixel features F^t and an off-the-shelf detector (Wang et al., 2023) Ⓐprovides bounding boxes. Unlike pipelines that crop to and predict poses for each box, CoMotion uses the detected boxes solely for track instantiation and deletion. Specifically, we check whether there is agreement between these bounding boxes and our current tracks Ⓓ and use any disagreement as a cue to instantiate new tracks. Agreement is quantified in terms of intersection-over-union (IOU). Any unmatched boxes indicate a new track that can be instantiated. Any unmatched tracks are candidates for deletion. The heart of CoMotion is formed by the update module Ⓒ, which refines the current set of tracks. The update module does not use the detected boxes, drawing instead on the full image context encoded in the feature tensor F^t.

Image encoder Ⓑ. The image encoder produces features F^t by means of a ConvNextV2-L backbone (Woo et al., 2023). We take features across intermediate stages of the network at four output scales and linearly project and sum to produce a combined feature tensor at 1/8th the input resolution of the image. Most of our experiments are performed at a 512x512 input resolution, so this yields a 64x64 feature grid. We concatenate this with a positional embedding to encode (x, y) coordinate locations and flatten the resulting features into a set of tokens. We apply an MLP to produce *key* and *value* tensors that will be used in the attention operations in subsequent steps.

Initialization Ⓕ. The initialization module Ⓕ accepts unmatched bounding boxes and outputs initial track states. Given a set of detections, we run an initialization stage to produce initial pose estimates that serve as the starting point for new tracks. More specifically, the output per bounding box is a track state which is defined by the set of SMPL parameters (translation, shape, and joint angles) and an additional latent feature vector used for a recurrent hidden state. We perform cross-attention to attend to the current set of image tokens to get all initial track states. Bounding boxes are encoded into a set of tokens with an MLP from which we then produce queries and attend to the per-pixel keys and values defined above. The result of this attention operation is added to the encoded box tokens and followed by residual transformer layers. We then apply an MLP decoder to produce initial predictions for the SMPL poses and hidden feature states.

Update module Ⓒ & Ⓖ. The update module Ⓒ & Ⓖ refines the current set of tracks. Cross-attention is used in the same manner as in the initialization stage, but instead of encoding bounding boxes into tokens we encode the per-track state. Each SMPL pose and hidden feature vector is passed through an encoder to produce tokens to which we apply the same routine as above. The output decoder in this case produces a new SMPL pose for each track as well as an update to each hidden state. We use a GRU to perform the recurrent hidden state update.

When encoding the current pose, we perform forward kinematics to get the 3D joint positions, and then use the provided intrinsics matrix K to project those keypoints to 2D pixel positions. We pass the original SMPL parameters, the 3D coordinates, the projected 2D locations, and the hidden state into the encoder. For training stability, we detach the gradient of the SMPL pose parameters before getting passed into the encoder, as such the only gradient signal that gets passed across time is through the hidden features.

This update is the core of CoMotion. It can take an arbitrary number of tracks as input along with the latest image tokens and determine the right changes to make to all people simultaneously. Over the course of training, the network learns to associate the correct visual evidence to each person.

There are a couple of practical differences between this approach and a more traditional detect-and-associate tracking method. First, as discussed above, the network can continue to make predictions on tracks with no visible content in the current frame. CoMotion is structurally predisposed to learn object permanence. Similarly, a partial or indirect observation that may not trigger a detection can be leveraged by the network to maintain and update a pose track, even in the presence of overwhelming occlusion. The network bears the responsibility for determining what information from the current set of pixels is relevant for each track, and can draw on the entire image for this purpose. For more architecture details, please refer to Appendix A.3.

Termination Ⓔ. We do not delete a track immediately if it does not align with the detected bounding boxes. We keep tracks alive and continue to update their poses. This requires that our model performs some degree of forecasting and motion modeling. This makes the tracker robust to brief occlusions, but is not guaranteed to produce coherent long-term motion. In practice, we end tracks after a fixed window of no matches for more than one second or after being out-of-bounds for more than half a second.

4 TRAINING

Our training procedure is simpler than the tracking setting deployed at test time. To train on a particular clip we sample a subset of the annotated people in the scene, and provide their bounding boxes as input at the first time step. The network then runs a single initialization pass followed by an update for each frame of the video. We supervise all predicted poses at each timestep as well as the outputs of the initialization module.

At no point do we instantiate new tracks or delete existing tracks as we unroll through a clip during training. This is a practical detail to support training on minibatches. We also never train on more than 8 simultaneous tracks, but find that the network generalizes to scenes with many more people.

4.1 DATASETS

We train on the pseudo-labeled versions of InstaVariety (Kanazawa et al., 2019), COCO (Lin et al., 2014), and MPII (Andriluka et al., 2014) provided by (Goel et al., 2023). These datasets provide single images that capture diverse scenes and challenging poses with approximate SMPL annotations.

On video, we train on BEDLAM (Black et al., 2023), which is a synthetic dataset with motion sourced from AMASS (Mahmood et al., 2019) that provides a large number of sequences with perfect 3D ground-truth.

Finally, we train on PoseTrack (Andriluka et al., 2018) and JRDB (Martin-Martin et al., 2021), which are both annotated with 2D keypoints. PoseTrack is an extension of MPII consisting of short clips, while JRDB offers mostly pedestrian data but longer sequences with nontrivial tracking interactions.

4.2 CURRICULUM

We train CoMotion in three stages, each utilizing a different mix of datasets, supervision strategies, and loss functions.

Stage 1 begins with single-image pretraining. We fine-tune an off-the-shelf masked auto-encoder (MAE) pretrained image encoder (Woo et al., 2023) to produce high-quality pose features. Since these features are independent per frame, we achieve more efficient training with larger batch sizes focusing exclusively on single images from the pseudo-labeled InstaVariety, COCO, and MPII datasets. The image encoder is frozen for all subsequent stages.

To optimize the model's SMPL predictions, we employ three loss functions: an $\mathcal{L}_1$-loss to minimize 2D projection error, another $\mathcal{L}_1$-loss for the root-normalized 3D joint position error, and an $\mathcal{L}_2$-loss for the difference in SMPL joint angles. Since the pseudo-labeled SMPL beta parameters are generally unreliable, we refrain from training directly on them, instead applying regularization to minimize the norm of predicted betas: $\mathcal{L}_{\beta\text{reg}} = \lambda\|\beta\|_2^2$ with $\lambda = 0.005$. Additionally, we use a keypoint heatmap loss (Cao et al., 2017) as an additional auxiliary training signal. The model is trained for 300K iterations, which take approximately 3 days on 8 V100 GPUs. By the end of this stage, the model is capable of jointly estimating the 3D poses of multiple people within a single image, given their bounding boxes.

Stage 2 extends training to multiple frames. We continue to train on individual images, but now introduce short, 4-frame video clips from BEDLAM. The same loss functions from Stage 1 are applied, with one key modification: the model is now directly supervised on the SMPL beta parameters using high-quality ground truth from BEDLAM. During this stage, we freeze the image encoder and train the weights of the initialization and update modules for an additional 300K iterations, which also take approximately 3 days. By the end of Stage 2, the model has learned to predict and track 3D poses of multiple people through a short video snippet with improved temporal consistency.

Stage 3 focuses on the model's ability to handle longer video sequences. We again train on BEDLAM but expand the clip length to 16 frames. Additionally, we train on 8-frame clips from PoseTrack and 16-frame clips from JRDB. Since the only available supervision for these datasets is 2D, we only apply a 2D loss on PoseTrack and JRDB. The model is fine-tuned for 150K iterations over 2 days, keeping the image backbone frozen. By the end of stage 3, the model exhibits improved stability when tracking 3D poses across longer sequences.

This training curriculum allows us to efficiently train a system to handle video by front-loading the initial visual feature learning on single images before the slower video training stage. Also, we can fit much larger batches during video training since we do not have to calculate gradients for or update the weights of the image encoder. And even though we only train on short half-second clips, the model behaves well when unrolled for longer sequences.

5 Experiments

No standard benchmark evaluates the full combination of characteristics that CoMotion was designed for: tracking 3D poses of multiple people through video. For quantitative comparisons to prior work, we resort to evaluating CoMotion on a subset of its capabilities at a time, "projecting" it to existing benchmarks that focus either on tracking stability (without evaluating 3D pose accuracy) or pose estimation on images (without evaluating tracking). We use established benchmarks for 2D and 3D pose estimation and, separately, tracking, and adhere to established evaluation protocols and metrics. See the appendix for controlled experiments on the model architecture, datasets, and the training curriculum.

Tracking. Standard tracking benchmarks do not typically evaluate pose accuracy, but rather metrics based on bounding-box IOU. For tracking evaluation, we run our full stack on a video from beginning to end. Detections are provided by an off-the-shelf model that operates on individual image frames. (The frames are padded and resized to 512x512.) Poses are unrolled through video and no cropping is done to individuals.

To evaluate tracking across long sequences in crowded scenes, we use PoseTrack21 (Doering et al., 2022). We use the evaluation code provided by Doering et al. (2022) to compute standard tracking metrics. We report Multi-Object Tracking Accuracy (MOTA) which is an aggregate score that penalizes for missed detections, false positives, and ID switches (100 is a perfect score). Other metrics such as IDF1 and ID precision and recall (IDP and IDR) communicate how well the network preserves a single tracked identity per person. The results are listed in Table 1(top). CoMotion substantially outperforms prior work, improving MOTA by 8% and IDF1 by 10% relative to the prior state of the art.

Table 1: **Tracking evaluation.** Performance of CoMotion and several baselines on PoseTrack21. We report results using the official evaluation code (top) as well as results after fixing a bug in the evaluation code that caused the ‘ignore’ regions to be handled incorrectly (bottom, marked with a †).

Method	PoseTrack21			
	MOTA↑	IDF1↑	IDP↑	IDR↑
TRMOT (Wang et al., 2020)	47.2	57.3	70.0	46.6
FairMOT (Zhang et al., 2021b)	56.3	63.2	81.0	51.8
CorrTrack + ReID (Doering et al., 2022)	52.0	66.5	72.4	61.4
Tracktor++ (Bergmann et al., 2019)	59.5	69.3	76.4	63.5
CoMotion (ours)	**64.3**	**76.3**	**83.0**	**70.5**
4DHumans (Goel et al., 2023) †	56.7	70.9	**87.1**	59.7
CoMotion (ours) †	**68.1**	**77.9**	86.7	**70.5**

Table 2: **PoseTrack18 vs. PoseTrack21.** The annotations in PoseTrack18 are drastically incomplete, penalizing methods that correctly detect and track people in the scene. Indeed, this inspired the creation of PoseTrack21 (Doering et al., 2022), which provides more complete annotations and was released as a direct replacement of PoseTrack18 (same images, more complete annotations). We provide results on PoseTrack18 for backward compatibility with Goel et al. (2023), but strongly recommend that all future work adopt PoseTrack21 instead. (* We rerun the authors’ code in order to report performance on PoseTrack21.)

Method	PoseTrack18			PoseTrack21						
	HOTA↑	IDs↓	MOTA↑	MOTA↑	IDF1↑	IDP↑	IDR↑	FP↓	FN↓	FPS↑
4DHumans (Goel et al., 2023)	57.8	382	61.4	–	–	–	–	–	–	-
4DHumans (reproduced)*	58.0	349	**61.8**	56.7	70.9	87.1	59.7	7817	50652	0.51
CoMotion *strict*	**58.2**	**232**	59.9	61.8	74.0	**89.1**	63.3	**6086**	45664	-
CoMotion	54.2	332	51.3	**68.1**	**77.9**	86.7	**70.5**	8617	**34438**	**8.34**

In the course of performing this evaluation and analyzing the results, we found a bug in the PoseTrack21 evaluation code. PoseTrack21 provides partial annotations accompanied by ‘ignore’ regions that mark parts of images in which ground-truth data is missing. The ‘ignore’ regions are crucial for evaluation because they signify that tracks predicted in these regions should not be regarded as false positives – they may be correct, but may not align with ground-truth annotations because annotations in these regions are known to be missing. The bug pertains to the handling of these ‘ignore’ regions and caused many detections in the ‘ignore’ regions to be incorrectly regarded as false positives. Table 1(bottom) reports results after this bug is fixed. (Here we can only benchmark CoMotion and a method that is reproduced in our environment, since it’s not possible to carry numbers over from prior literature.) We will release our fix to PoseTrack21 and recommend that future work adopts the corrected evaluation code. More details can be found in the appendix.

The most competitive baseline is 4D Humans (Goel et al., 2023). The paper of Goel et al. (2023) reports results on PoseTrack18 (Andriluka et al., 2018) but not on the more accurately and completely annotated PoseTrack21. Unfortunately, due to the drastically incomplete annotations in PoseTrack18 (which motivated the creation of PoseTrack21), we observe that tracking evaluation on PoseTrack18 can be substantially misleading. Many sequences in PoseTrack18 only have annotations for a fraction of the people in the scene, and models are penalized for *correctly* detecting and tracking people who lack annotations. See the appendix for examples.

Since 4D Humans does not report results on PoseTrack21, we rerun their method and reproduce their original predictions (which match the numbers reported in their paper on their PoseTrack18 evaluation). In order to do well on the incomplete PoseTrack18 setting, it appears that 4D Humans is calibrated to produce fewer detections to avoid getting incorrectly penalized for tracking people who were not annotated in the dataset. This results in potentially lower MOTA on PoseTrack21 than they might achieve had they calibrated their system differently. We dig further into this in Table 2, observing that we can match their numbers in the original PoseTrack18 setting by applying a stricter threshold on bounding box detection scores (0.7 instead of 0.3) to preserve tracks and dismissing a larger number of (correct) detections.

Table 3: **Pose estimation.** Normalized PCK accuracy on projected 2D keypoints at varying thresholds on the COCO and PoseTrack datasets, alongside MPJPE of 3D keypoints on the 3DPW dataset. As noted by Goel et al. (2023), the 3DPW metrics can be anti-correlated with performance in the wild. See text for analysis.

Method	COCO		PoseTrack		3DPW	
	PCKn@0.05↑	PCKn@0.1↑	PCKn@0.05↑	PCKn@0.1↑	MPJPE↓	PA-MPJPE↓
PyMAF (Zhang et al., 2021a)	0.68	0.86	0.77	0.92	92.8	58.9
CLIFF (Li et al., 2022)	0.64	0.88	0.75	0.92	**69.0**	**43.0**
PARE (Kocabas et al., 2021)	0.72	0.91	0.79	0.93	82.0	50.9
PyMAF-X (Zhang et al., 2023a)	0.79	0.93	0.85	0.95	78.0	47.1
HMR 2.0a (Goel et al., 2023)	0.79	0.95	0.86	0.97	70.0	44.5
HMR 2.0b (Goel et al., 2023)	**0.86**	**0.96**	**0.90**	**0.98**	81.3	54.3
CoMotion	0.84	**0.96**	**0.90**	**0.98**	77.2	51.2

When evaluated on PoseTrack21 we observe that our ‘strict’ model is better than 4D Humans across the board. When we replace the strict threshold with our default, the system is heavily penalized in PoseTrack18, since many correct tracks are regarded as false positives, lowering MOTA from 59.9 to 51.3. But when these *same predictions* are evaluated on the more complete annotations provided by PoseTrack21, MOTA *increases* from 61.8 to 68.1.

Pose estimation. Pose estimation is typically assessed in an ‘oracle’ single-person setting where the bounding box of the target individual is known in advance. To compare to pose estimation methods using standard pose estimation metrics, CoMotion is constrained to operate within single frames and tight bounding-box crops. We evaluate both 2D and 3D metrics, since 2D pose datasets offer more diverse in-the-wild data with challenging poses. Specifically, we report Percentage of Correct Keypoints (PCK) on COCO (Lin et al., 2014) and PoseTrack18 (Andriluka et al., 2018), and Mean Per-Joint Position Error (MPJPE) on 3DPW (von Marcard et al., 2018). PCK calculates the percentage of estimates whose distance to the ground truth falls under a given threshold, while MPJPE is the mean distance between 3D points after centering around the pelvis. PA-MPJPE performs an additional Procrustes Alignment step first. The results are summarized in Table 3. We follow the same setup as prior work, with the same bounding boxes, cropping, and resizing of each subject. Note that no video information is used here, so CoMotion is restricted to a subset of its capability. In this constrained single-image pose estimate regime, we perform on par with the state-of-the-art HMR 2.0b model (Goel et al., 2023).

As noted by Goel et al. (2023), 3DPW metrics are not necessarily correlated to in-the-wild performance. We tuned CoMotion for performance in the wild and will release the model upon publication.

Analysis. CoMotion substantially outperforms the state of the art on multi-person tracking, while also yielding state-of-the-art 3D pose estimates. At the same time, CoMotion is an order of magnitude faster than 4DHumans, the strongest comparable system in the literature (Goel et al., 2023). Beyond the numerical results, there are substantial qualitative differences in the behavior of CoMotion and 4DHumans. On challenging or truncated poses, 4DHumans sometimes exhibits high-frequency, abrupt changes as it jumps between possible interpretations of a given track from frame to frame. In contrast, CoMotion is more temporally coherent due to recurrently unrolling predictions across time (Figure 3). See the appendix and the supplement for additional analysis and results.

5.1 CONTROLLED EXPERIMENTST

We conduct several controlled experiments on architecture and dataset decisions and only report key results here due to space constraints. For a more thorough overview of these results we refer the reader to Appendix A.2.

Architecture: We test how the model performs when removing the initialization module and replacing the initialization with a heuristic placement in camera space conditioned on the provided bounding box and a default SMPL pose. The network is able to learn to update this initial estimate, but keypoint localization accuracy drops by 22%. We also test removing the recurrent hidden features. In this case, pose accuracy is not too adversely affected, but we see ID switches increase by about 25%.

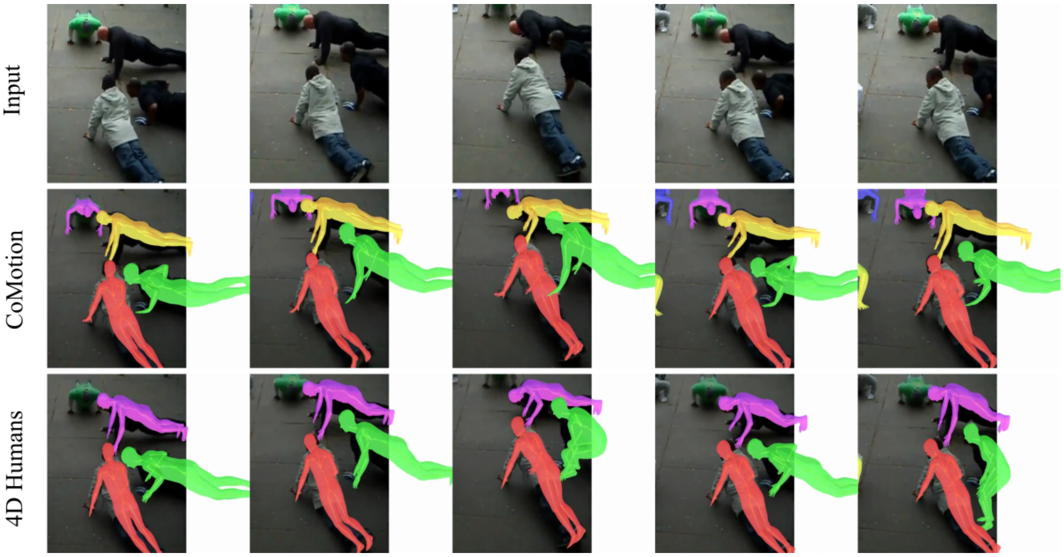

Figure 3: We compare predictions made by CoMotion and 4D Humans unrolled through time on a sample from PoseTrack. With no temporal consistency we see that 4D Humans occasionally makes abrupt changes to the estimated pose in this challenging setting where most of the person is truncated out the right side of the frame.

Training setting: We further analyze how training datasets and curriculum affect performance. If we leave out BEDLAM (Table 5), the performance on PoseTrack is nearly the same as it measures only 2D accuracy, but 3D performance drops significantly on 3DPW. In addition, we find that the model benefits from the full multi-stage training curriculum. In particular, training on longer sequences (16 frames as opposed to 4) yields improvements. We expect training on much longer temporal windows to lead to even more robust tracking behavior, but leave this exploration to further work due to substantial increase in required computational resources.

6 Conclusion

CoMotion performs joint multi-person 3D pose tracking from monocular video. It operates online, processing each incoming frame in a streaming fashion. By unrolling predictions across time and allowing the network to jointly attend to all pose tracks and the full image context, CoMotion maintains accurate and temporally stable tracking and robust inference through occlusion.

Many characteristics of the system can benefit from scaling, including model size, training time, and input resolution. A significant impediment to progress is the lack of high-quality video training data. Given the beneficial effect of large-scale pseudo-labeled single-image datasets (Goel et al., 2023), we believe that the community would benefit from analogous efforts in the video domain.

Beyond scale and additional training data, a number of architectural ideas may benefit the system as well. For example, detection could be treated jointly and trained end-to-end with pose estimation and tracking. Recent work such as MultiHMR (Baradel et al., 2024) suggests that the basic cross-attention formulation is well-suited to joint detection and pose estimation.

Another promising direction is to also model camera motion. An emerging body of work models 3D poses in a consistent world coordinate frame (Shin et al., 2023; Wang et al., 2024; Kocabas et al., 2024; Yin et al., 2024). We believe that decoupling the camera from the motion of people in the scene can increase robustness to extreme camera motion. Furthermore, CoMotion does not currently reason about absolute scale, thus 3D poses are not grounded in a shared extrinsic world frame. This is most salient with children, who are not modeled well by the SMPL parameterization and are commonly placed further into the distance as larger adult-sized humans.

CoMotion provides a clean high-performing starting point for exploring these and other ideas. We will release the implementation and the trained model upon publication.

REFERENCES

Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2D human pose estimation: New benchmark and state of the art analysis. In *CVPR*, 2014.

Mykhaylo Andriluka, Umar Iqbal, Eldar Insafutdinov, Leonid Pishchulin, Anton Milan, Juergen Gall, and Bernt Schiele. PoseTrack: A benchmark for human pose estimation and tracking. In *CVPR*, 2018.

Fabien Baradel, Matthieu Armando, Salma Galaaoui, Romain Brégier, Philippe Weinzaepfel, Grégory Rogez, and Thomas Lucas. Multi-HMR: Multi-person whole-body human mesh recovery in a single shot. In *ECCV*, 2024.

Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *ICCV*, 2019.

Michael J Black, Priyanka Patel, Joachim Tesch, and Jinlong Yang. BEDLAM: A synthetic dataset of bodies exhibiting detailed lifelike animated motion. In *CVPR*, 2023.

Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J Black. Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image. In *ECCV*, 2016.

Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017.

Andreas Doering, Di Chen, Shanshan Zhang, Bernt Schiele, and Juergen Gall. PoseTrack21: A dataset for person search, multi-object tracking and multi-person pose tracking. In *CVPR*, 2022.

Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. RMPE: Regional multi-person pose estimation. In *ICCV*, 2017.

Ruopeng Gao and Limin Wang. MeMOTR: Long-term memory-augmented transformer for multi-object tracking. In *ICCV*, 2023.

Rohit Girdhar, Georgia Gkioxari, Lorenzo Torresani, Manohar Paluri, and Du Tran. Detect-and-track: Efficient pose estimation in videos. In *CVPR*, 2018.

Shubham Goel, Georgios Pavlakos, Jathushan Rajasegaran, Angjoo Kanazawa, and Jitendra Malik. Humans in 4D: Reconstructing and tracking humans with transformers. In *ICCV*, 2023.

Shaoli Huang, Mingming Gong, and Dacheng Tao. A coarse-fine network for keypoint localization. In *ICCV*, 2017.

Eldar Insafutdinov, Mykhaylo Andriluka, Leonid Pishchulin, Siyu Tang, Evgeny Levinkov, Bjoern Andres, and Bernt Schiele. ArtTrack: Articulated multi-person tracking in the wild. In *CVPR*, 2017.

Angjoo Kanazawa, Michael J Black, David W Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *CVPR*, 2018.

Angjoo Kanazawa, Jason Y Zhang, Panna Felsen, and Jitendra Malik. Learning 3D human dynamics from video. In *CVPR*, 2019.

Rawal Khirodkar, Aayush Bansal, Lingni Ma, Richard Newcombe, Minh Vo, and Kris Kitani. Ego-humans: An ego-centric 3D multi-human benchmark. In *ICCV*, 2023.

Rawal Khirodkar, Timur Bagautdinov, Julieta Martinez, Su Zhaoen, Austin James, Peter Selednik, Stuart Anderson, and Shunsuke Saito. Sapiens: Foundation for human vision models. In *ECCV*, 2024.

Muhammed Kocabas, Nikos Athanasiou, and Michael J Black. VIBE: Video inference for human body pose and shape estimation. In *CVPR*, 2020.

Muhammed Kocabas, Chun-Hao P Huang, Otmar Hilliges, and Michael J Black. PARE: Part attention regressor for 3D human body estimation. In *ICCV*, 2021.

Muhammed Kocabas, Ye Yuan, Pavlo Molchanov, Yunrong Guo, Michael J Black, Otmar Hilliges, Jan Kautz, and Umar Iqbal. PACE: Human and camera motion estimation from in-the-wild videos. In *3DV*, 2024.

Zhihao Li, Jianzhuang Liu, Zhensong Zhang, Songcen Xu, and Youliang Yan. CLIFF: Carrying location information in full frames into human pose and shape estimation. In *ECCV*, 2022.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014.

Huan Liu, Qiang Chen, Zichang Tan, Jiang-Jiang Liu, Jian Wang, Xiangbo Su, Xiaolong Li, Kun Yao, Junyu Han, Errui Ding, et al. Group pose: A simple baseline for end-to-end multi-person pose estimation. In *ICCV*, 2023.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. SMPL: A skinned multi-person linear model. *TOG*, 34(6), 2015.

Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. AMASS: Archive of motion capture as surface shapes. In *ICCV*, 2019.

Roberto Martin-Martin, Mihir Patel, Hamid Rezatofighi, Abhijeet Shenoi, JunYoung Gwak, Eric Frankel, Amir Sadeghian, and Silvio Savarese. JRDB: A dataset and benchmark of egocentric robot visual perception of humans in built environments. *TPAMI*, 45(6), 2021.

Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. TrackFormer: Multi-object tracking with transformers. In *CVPR*, 2022.

George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In *CVPR*, 2017.

Georgios Pavlakos, Luyang Zhu, Xiaowei Zhou, and Kostas Daniilidis. Learning to estimate 3D human pose and shape from a single color image. In *CVPR*, 2018.

Jathushan Rajasegaran, Georgios Pavlakos, Angjoo Kanazawa, and Jitendra Malik. Tracking people by predicting 3D appearance, location & pose. In *CVPR*, 2022.

N Dinesh Reddy, Laurent Guigues, Leonid Pishchulin, Jayan Eledath, and Srinivasa G Narasimhan. TesseTrack: End-to-end learnable multi-person articulated 3D pose tracking. In *CVPR*, 2021.

Dahu Shi, Xing Wei, Liangqi Li, Ye Ren, and Wenming Tan. End-to-end multi-person pose estimation with transformers. In *CVPR*, 2022.

Soyong Shin, Juyong Kim, Eni Halilaj, and Michael J Black. Wham: Reconstructing world-grounded humans with accurate 3D motion. *arXiv:2312.07531*, 2023.

Peize Sun, Yi Jiang, Rufeng Zhang, Enze Xie, Jinkun Cao, Xinting Hu, Tao Kong, Zehuan Yuan, Changhu Wang, and Ping Luo. TransTrack: Multiple-object tracking with transformer. *arXiv:2012.15460*, 2020.

Yu Sun, Qian Bao, Wu Liu, Tao Mei, and Michael J Black. TRACE: 5D temporal regression of avatars with dynamic cameras in 3D environments. In *CVPR*, 2023.

Zhenhua Tang, Zhaofan Qiu, Yanbin Hao, Richang Hong, and Ting Yao. 3D human pose estimation with spatio-temporal criss-cross attention. In *CVPR*, 2023.

Edward Vendrow, Duy Tho Le, Jianfei Cai, and Hamid Rezatofighi. Jrdb-pose: A large-scale dataset for multi-person pose estimation and tracking. In *CVPR*, 2023.

Timo von Marcard, Roberto Henschel, Michael J Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3D human pose in the wild using IMUs and a moving camera. In *ECCV*, 2018.

Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *CVPR*, 2023.

Yufu Wang and Kostas Daniilidis. ReFit: Recurrent fitting network for 3D human recovery. In *ICCV*, 2023.

Yufu Wang, Ziyun Wang, Lingjie Liu, and Kostas Daniilidis. Tram: Global trajectory and motion of 3D humans from in-the-wild videos. *arXiv:2403.17346*, 2024.

Zhongdao Wang, Liang Zheng, Yixuan Liu, Yali Li, and Shengjin Wang. Towards real-time multi-object tracking. In *ECCV*, 2020.

Ross Wightman. Pytorch image models. `https://github.com/huggingface/pytorch-image-models`, 2019.

Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. ConvNeXt V2: Co-designing and scaling convnets with masked autoencoders. In *CVPR*, 2023.

Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *ECCV*, 2018.

Wanqi Yin, Zhongang Cai, Ruisi Wang, Fanzhou Wang, Chen Wei, Haiyi Mei, Weiye Xiao, Zhitao Yang, Qingping Sun, Atsushi Yamashita, et al. WHAC: World-grounded humans and cameras. *arXiv:2403.12959*, 2024.

Yingxuan You, Hong Liu, Ti Wang, Wenhao Li, Runwei Ding, and Xia Li. Co-evolution of pose and mesh for 3D human body estimation from video. In *ICCV*, 2023.

Fangao Zeng, Bin Dong, Yuang Zhang, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. MOTR: End-to-end multiple-object tracking with transformer. In *ECCV*, 2022.

Hongwen Zhang, Yating Tian, Xinchi Zhou, Wanli Ouyang, Yebin Liu, Limin Wang, and Zhenan Sun. PyMAF: 3D human pose and shape regression with pyramidal mesh alignment feedback loop. In *ICCV*, 2021a.

Hongwen Zhang, Yating Tian, Yuxiang Zhang, Mengcheng Li, Liang An, Zhenan Sun, and Yebin Liu. PyMAF-X: Towards well-aligned full-body model regression from monocular images. *TPAMI*, 45 (10), 2023a.

Jinlu Zhang, Zhigang Tu, Jianyu Yang, Yujin Chen, and Junsong Yuan. MixSTE: Seq2seq mixed spatio-temporal encoder for 3D human pose estimation in video. In *CVPR*, 2022.

Juze Zhang, Ye Shi, Yuexin Ma, Lan Xu, Jingyi Yu, and Jingya Wang. IKOL: Inverse kinematics optimization layer for 3D human pose and shape estimation via gauss-newton differentiation. In *AAAI*, 2023b.

Siwei Zhang, Bharat Lal Bhatnagar, Yuanlu Xu, Alexander Winkler, Petr Kadlecek, Siyu Tang, and Federica Bogo. Rohm: Robust human motion reconstruction via diffusion. In *CVPR*, 2024.

Yi Zhang, Pengliang Ji, Angtian Wang, Jieru Mei, Adam Kortylewski, and Alan Yuille. 3D-aware neural body fitting for occlusion robust 3D human pose estimation. In *ICCV*, 2023c.

Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. FairMOT: On the fairness of detection and re-identification in multiple object tracking. *IJCV*, 129(11), 2021b.

Qitao Zhao, Ce Zheng, Mengyuan Liu, Pichao Wang, and Chen Chen. PoseFormerV2: Exploring frequency domain for efficient and robust 3D human pose estimation. In *CVPR*, 2023.

Ce Zheng, Sijie Zhu, Matias Mendieta, Taojiannan Yang, Chen Chen, and Zhengming Ding. 3D human pose estimation with spatial and temporal transformers. In *ICCV*, 2021.

A Appendix

A.1 Tracking Evaluation Details

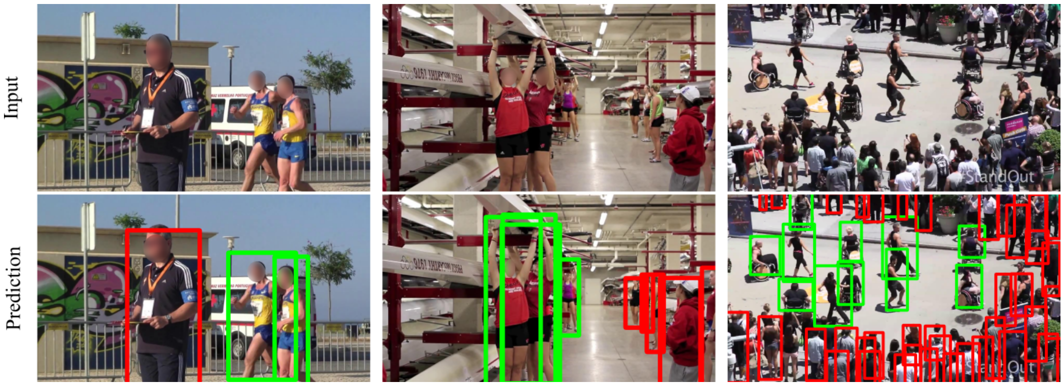

Figure 4: **Incorrect handling of missing annotations in PoseTrack18.** Due to incomplete annotations in PoseTrack18, tracks may be incorrectly regarded as "false positives". We show representative samples where annotations are green and "false positives" are red.

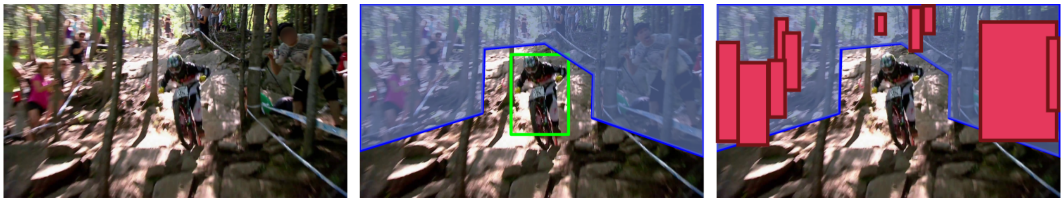

Figure 5: **Incorrect handling of missing annotations in PoseTrack 21.** PoseTrack21 addresses the incompleteness of PoseTrack18 annotations by providing 'ignore' regions to accompany the annotated tracks. For the frame on the left, the center image illustrates the ground truth annotation, which consists of the annotation of the person in the center (shown in green) and a polygon defining the 'ignore' region in blue. The right image shows predicted tracks in red, which are still penalized as false positives by the PoseTrack21 evaluation code despite being contained in the 'ignore region'. This is a bug that we fix.

Tracking evaluation metrics are sensitive to false positives. When calculating MOTA a perfect score is a 100.0, but there's no bound to how negative a score may be when penalized for false positives. For example, we observe on the scene illustrated in Figure 5, if false positives are not ignored our method garners a MOTA of around -1500.

Figure 4 visualizes the problematic handling of missing annotations in PoseTrack18. Due to missing annotations, correctly predicted tracks are labeled as false positives, which may produce misleading results.

To address this problem, Doering et al. (2022) introduced PoseTrack21 with more comprehensive annotations. PoseTrack21 provides keypoints for many more people and marks regions with missing annotations through "ignore regions". Unfortunately, the provided evaluation code contains a bug that still allows for incorrect handling of missing annotations. Figure 5 visualizes the behavior of the PoseTrack21 evaluation. Akin to PoseTrack18, predicted boxes in the ignore region are treated as false positives.

The issue has to do with the calculation determining whether or not to ignore a box. The evaluation code looks at the IOU between the ignore region and the box and discards any box with an IOU greater than 0.1. Unfortunately, when the ignore region is particularly large, most boxes do not intersect a large percentage of the region's area. The IOUs in the example above vary between 0.01 and 0.07 – not enough to be discarded.

We make a one line change to the PoseTrack21 evaluation code for our "fixed" evaluation: instead of IOU, we calculate the percentage of the bounding box that overlaps with the ignore region. This results in the expected evaluation behavior on scenes with ignore regions. We adjust the discard threshold from 0.1 to 0.3, so this means if 30% of a bounding box overlaps into an ignore region it will be discarded.

There are a few minor quirks that still remain in the annotations and evaluation. In some clips, ignore regions appear and disappear, and occasionally completely overlap with ground-truth annotations. We observe in the evaluation code by the PoseTrack21 authors that there is a mechanism to recover predictions that are matched to ground-truth tracks that have been filtered out by the ignore region. Perhaps this is to address the samples that have been both marked as ground-truth boxes and covered in ignore regions. Overall, these present a modest amount of corner cases and do not adversely affect the evaluation in the same way as the ignore region IOU calculation.

A.2 Controlled experiments

We perform additional experiments to investigate the impact of major design decisions on the final performance of the proposed approach. We report results on the PoseTrack validation set, on the 3DPW validation set, and on a subset of clips from EgoHumans (Khirodkar et al., 2023).

With EgoHumans, we do not follow an official evaluation setting, but create our own evaluation by sampling some of the exocentric camera clips from the Lego, Tagging, and Fencing scenes. We find EgoHumans interesting as one of the few datasets with 3D annotations and complete annotations for multiple people. Critically, there are annotations through occlusion because data was collected in a multi-view setting. Most other evaluation datasets specifically do not have ground truth for occluded and truncated samples.

All evaluation is done across video in the multi-person setting with full images resized and padded to 512x512. For the purpose of this evaluation we do not use the tracking stack but provide boxes for the target people we want predictions on in frame 0 and we then unroll the model through the full sequence without adding or removing any tracks. On PoseTrack, we unroll 16 frames, and on 3DPW and EgoHumans we unroll 48 frames.

One note, for these ablations, the 2D PCK @ 0.05 on PoseTrack is calculated differently from the numbers reported in Table 3. It is more strict and not comparable to the values in the main paper. We use this metric only in the context of ablations comparing different versions of our model.

A.2.1 Architecture

We assess the importance of our proposed pose initialization and recurrent hidden features through an ablation study. We report detailed metrics in Table 4. The first row represents our full approach, the second row removes the hidden state, and the third row alternatively replaces the initialization module with a simple heuristic.

Hidden state. Without a hidden state, the only information passed across consecutive frames are the previous SMPL poses. For the vast majority of tracking situations this is sufficient to know which pose corresponds to which person, but can lead to failures in tricky corner cases with similar poses in similar positions. We observe that without a hidden state for each track overall performance on challenging videos from PoseTrack drops, accompanied by a notable decrease in tracking performance. Removing the hidden features increases ID switches by 15%.

Initialization module. We replace the initialization module with a simple heuristic that requires no computation or learned parameters. From a bounding box, we define a fixed procedure to produce an initial root position and default SMPL pose. This still provides a valid conditioning signal that the update module can learn to refine. To compensate for the lost model capacity, we add more compute to the update step. We find it is more helpful to provide a higher quality learned pose from the initialization module. We observe that removing the initialization module results in a 22% reduction in pose accuracy.

Table 4: **Architecture ablations.** From our full system (first row) we remove alternatively the hidden state (second row) or replace the pose initialization by a simple heuristic (third row). Removing the hidden state worsens tracking performance on PoseTrack. Removing the pose initialization module causes an even larger drop in tracking performance.

Initialization	Hidden	PoseTrack				EgoHumans		3DPW	
		2D@0.05↑	MOTA↑	IDF1↑	IDs↓	3D@0.1↑	MPJPE↓	3D@0.1↑	MPJPE↓
✓	✓	**66.92**	**67.2**	**77.4**	**459**	**59.69**	111.41	81.00	77.05
✓	-	65.49	63.8	75.0	526	58.29	119.57	**82.29**	**74.56**
-	✓	52.22	55.6	68.7	681	57.21	**108.65**	77.24	77.61

A.2.2 Effect of adding 3D video supervision

We investigate the role of 3D video supervision through BEDLAM (Black et al., 2023) in the performance of our model. Table 5 reports results of our full model and without training on BEDLAM. While removing BEDLAM marginally affects CoMotion's 2D performance on PoseTrack and 3DPW, we observe a strong effect on 3D performance on EgoHumans and 3DPW. This suggests that in order to maintain high quality 3D poses while unrolling through video it is important to include some 3D video supervision.

Table 5: Ablation study on the benefit of synthetic data (BEDLAM).

BEDLAM	PoseTrack	EgoHumans			3DPW		
	2D@0.05↑	2D@0.05↑	3D@0.1↑	MPJPE↓	2D@0.05↑	3D@0.1↑	MPJPE↓
✓	**66.92**	**67.31**	**59.69**	**111.41**	91.29	**81.00**	**77.05**
	66.04	63.81	48.98	133.58	**92.44**	72.84	92.72

A.2.3 Curriculum

We examine the decision to train the model in three stages (Table 6) while increasing the sequence length. A model trained only on Stage 1 performs poorly, as it has only been exposed to single images and lacks the ability to update poses across different frames (all of the evaluation in this table is across 16-48 frames). With the inclusion of Stage 2, the model's performance significantly improves across all metrics. Stage 3 training further enhances the results. We also experiment with different sequence lengths during Stage 3. Detailed results are included in Table 6.

Table 6: Ablation study on the training curriculum.

Stage 1	Stage 2	Stage 3	PoseTrack	EgoHumans			3DPW		
			2D@0.05	2D@0.05	3D@0.1	MPJPE	2D@0.05	3D@0.1	MPJPE
✓			26.20	12.98	9.29	320.62	27.90	18.17	243.64
✓	✓		52.29	59.61	57.53	121.03	90.50	**81.99**	75.49
✓	✓	✓(4f)	66.71	65.02	55.50	117.30	91.00	81.62	**73.66**
✓	✓	✓(16f)	**66.92**	**67.31**	**59.69**	**111.41**	**91.29**	81.00	77.05

A.2.4 Runtime

We measure the runtimes of CoMotion and baselines on the PoseTrack21 validation set and report results in Figure 6. All measurements were made on the same hardware using the code provided by the authors of each method. We measure the time to run the complete tracking stack unrolled across all PoseTrack validation videos. We find that CoMotion significantly outperforms prior work. Specifically, CoMotion is approximately 2x faster than PARE (123ms vs 258ms) and 17x faster than 4D Humans (123ms vs 2163ms) on average.

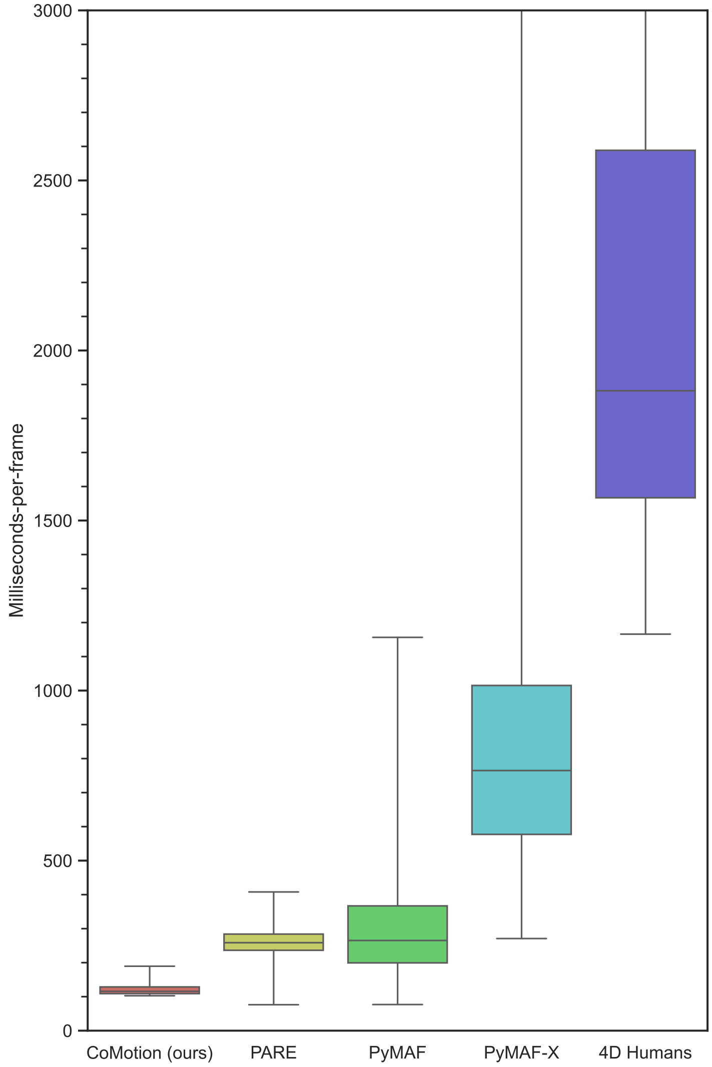

Figure 6: Comparing the per-frame runtime of CoMotion with prior work on the PoseTrack21 validation set. All measurements were made on a V100 GPU using the code released by the respective authors. CoMotion is significantly faster than prior work. Specifically, CoMotion is approximately 2x faster than PARE (123ms vs 258ms) and 17x faster than 4D Humans (123ms vs 2163ms) on average.

A.2.5 SENSITIVITY TO PROVIDED CAMERA INTRINSICS.

To assess the sensitivity of our model to the provided camera intinsics, we evaluated CoMotion on PoseTrack videos (for which we do not have ground truth camera calibration) and varied the camera intrinsics. Figure 7 visualizes the results. We find that the network's 2D keypoint accuracy remains consistently high across a wide range of input focal lengths.

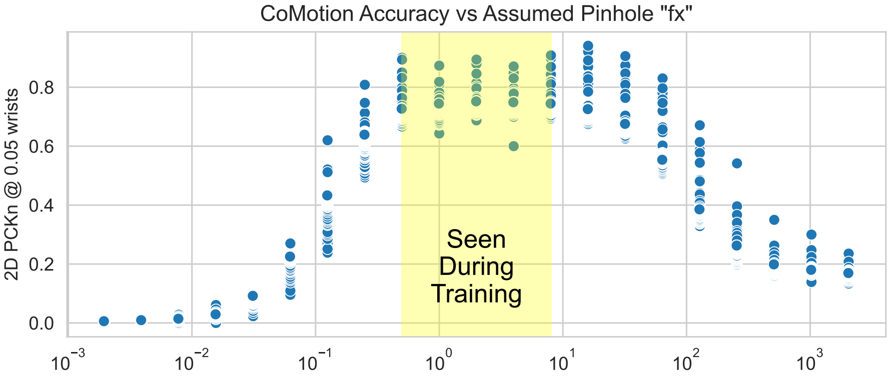

Figure 7: **Analysis on assumed focal length:** To investigate how the focal length of the intrinsics matrix affects performance we run our model on PoseTrack videos (for which we do not have ground truth camera calibration) and report 2D PCK accuracy. We adjust the assumed focal length and observe that the network's 2D keypoint accuracy is consistent as long as we remain within a realm of values which correspond to what one would typically find with most camera hardware apart from extremely wide-angle options such as a fish-eye lens.

A.3 ARCHITECTURE DETAILS

Image encoder: We use the ConvNextV2 (Woo et al., 2023) implementation provided in the timm library (Wightman, 2019). As with most convolutional backbones, features are extracted in consecutive stages to progressively lower resolutions. There are four stages yielding features at 1/4th, 1/8th, 1/16th, and 1/32nd of the input image resolution. As a simple strategy to fuse early and late features, we linearly project the outputs from each stage to a feature tensor at 1/8th the input resolution. For example, a 2x2 convolution with stride 2 is used on the highest resolution features, while a 1x1 convolution is applied to the lower resolution feature maps with an output of either 4 or 16 times the number of feature channels which can be reshaped into an upsampled tensor. The resulting features are added together to produce a final tensor upon which the model will perform cross-attention. Each feature vector at each spatial location now serves as one token to be attended to. We use standard sinusoidal positional embeddings to encode the x and y location of each vector as additional context. Here we provide PyTorch-like pseudo-code for producing the image keys and values:

```
def get_image_tokens(image):
    # Run ConvNextV2 stages
    x = convnext.stem(image)
    feat_pyramid = []
    for stage in convnext.stages:
        x = stage(x)
        # Linearly project image features from each stage
        feat_pyramid.append(conv(x))

    # Upsample low-res features (using einops notation)
    upsample_pattern = "... (c h2 w2) h w -> ... c (h h2) (w w2)"
    feat_pyramid[2].rearrange(upsample_pattern, h2=2, w2=2)
    feat_pyramid[3].rearrange(upsample_pattern, h2=4, w2=4)

    # Fuse into single tensor at 1/8th input resolution
    px_feats = sum(feat_pyramid)
    px_feats = layer_norm(px_feats)
    px_feats = cat([px_feats, positional_encoding], 1))

    # Flatten and rearrange into tokens
    px_feats.rearrange("... c h w -> ... (h w) c")
    image_key = linear(px_feats)
    image_value = linear(px_feats)

    return image_key, image_value
```

Cross-attention: We perform cross-attention in the same manner both during the initialization and update stages. First, we encode inputs into a set of tokens to be operated on. This would either be the bounding boxes or the current pose estimates and hidden state. Specifically, we apply two linear layers with an intermediate layer normalization and GELU activation. This MLP maps from the input dimension to an output dimensionality of $M * C$ which is rearranged to M tokens of dimensionality C. This is computed separately per-person resulting in a set of tokens with the shape NxMxC where N is the number of people.

Linear layers are applied to the pose tokens to get queries. From there, we perform cross-attention with the image tokens and apply an MLP to the result to compute a residual update to the pose tokens. We then apply a set of transformer layers to the pose tokens. These layers are applied in two stages. To reduce the overhead of self-attention across the union of all tokens, we decompose the attention operations such that they are applied first over the person dimension and then separately over the token dimension. Finally, MLPs are used to decode the relevant outputs from the updated tokens. For more model capacity, we instantiate and apply multiple rounds of cross-attention layers. For an overview of these operations, we recommend referring to the pseudo-code provided below.

```
def cross_attention(image_key, image_value, tokens):
    # Image key and value shape
    # B: batch_size, H*W: height x width, C: feature_dim
    # Input tokens shape
    # B: batch_size, N: num_people, M: num_tokens, C: feature_dim

    # Perform cross attention on union of all tokens for all people
    q = token_to_query(tokens)
    q.rearrange("b n m c -> b (n m) c")
    px_feedback = F.scaled_dot_product_attention(
        q, image_key, image_value
    )
    px_feedback.rearrange("b (n m) c -> b n m c")
    tokens = tokens + post_attention_mlp(px_feedback)

    # Attention between people
    tokens.rearrange("b n m c -> (b m) n c")
    tokens = cross_people_attention(tokens)

    # Attention per-person
    tokens.rearrange("(b m) n c -> (b n) m c")
    tokens = per_person_attention(tokens)
    tokens.rearrange("(b n) m c -> b n m c")

    return tokens

def initialization(K, image_key, image_value, bbox):
    # Encode tokens
    tokens = encode_bbox_to_tokens(bbox)

    # Apply cross-attention
    tokens = cross_attention(tokens)
    tokens = cross_attention(tokens)

    # Decode outputs
    smpl_params = decode_pose(tokens)
    smpl_params = adjust_trans(K, smpl_params)
    hidden = decode_hidden(tokens)
    return smpl_params, hidden

def update(K, image_key, image_value, smpl_params, hidden):
    # Encode tokens
    pred_2d = get_2d_projection(K, smpl_params)
    tokens = encode_2d_to_tokens(pred_2d)
    tokens += encode_smpl_to_tokens(smpl_params)
    tokens += encode_hidden_to_tokens(hidden)

    # Apply cross-attention
    tokens = cross_attention(tokens)
    tokens = cross_attention(tokens)

    # Decode outputs
    smpl_params = decode_update(tokens, smpl_params)
    hidden = gru_update(tokens, hidden)
    return smpl_params, hidden
```

A.4 ADDITIONAL TRAINING DETAILS

Intrinsics: We take measures to ensure the model's predictions align with the provided input intrinsics matrix K. During initialization, instead of directly predicting a distance from the camera, the network predicts a value ρ that is proportional to inverse depth. We use an exponential to ensure the value is nonnegative. This is then multiplied by the provided focal length to give us an initial depth estimate. The x, y values are predicted in pixel space (u,v) and mapped to an initial SMPL translation estimate γ_0 with the following steps:

$$z = \frac{f_x}{e^{\rho}} \tag{1}$$

$$\gamma_0 = z \cdot \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{2}$$

During training, we have two settings that occur in our datasets. Either ground-truth intrinsics are available (for example on a synthetic dataset like BEDLAM) or the images are sourced from the internet and thus we have no knowledge of the corresponding calibration data. In the latter case, we randomly sample focal lengths and provided a made-up intrinsics matrix to accompany training samples. This encourages sensible behavior regardless of the input matrix (Figure 7).

SMPL parameter update: For the update step, we investigated whether to perform a residual update to the current SMPL parameters or to predict new parameters. In the end, we use a mix of both:

- γ *(translation):* We find training to be more stable with a residual update to the translation term. Note, we enforce a positive depth term at initialization since we only supervise the model to initialize visible people in front of the camera. But this restriction does not hold for subsequent updates. The network could potentially model situations where the person falls behind the camera plane. Our datasets do not include many situations where this would be annotated so it is not a case that we expect the model to handle particularly well and we end tracks that fall out-of-bounds in such a manner.
- θ *(pose):* As for the pose term, we find we achieve better test time behavior when the network predicts a new pose each timestep. While training does proceed well when predicting a residual update, an unfortunate consequence is at test time the network steers towards odd behavior across long temporal sequences. It has a tendency to accumulate updates into out-of-distribution joint angles from which it cannot recover. If the network instead predicts a new pose each step, we do not run into this issue.
- β *(shape):* We determined it is better to not update the betas which determine body shape. When allowed to do so, the network would occasionally opt to make a person larger or smaller instead of physically moving them towards or away from the camera. Results were more sensible and stable with a fixed body shape across time.

In summary, the update module produces the following outputs: ($\Delta\gamma_{pred}$, θ_{pred}). And we define $\gamma_t = \gamma_{t-1} + \Delta\gamma_{pred}$, $\theta_t = \theta_{pred}$, and $\beta_t = \beta_0$.

Handling multiple annotation modalities: We integrate several datasets with different formats for human body pose annotations. The mismatch in formats ranges from different granularity of keypoints (e.g., COCO (Lin et al., 2014) provides face keypoints (eyes, ears, nose) while MPII (Andriluka et al., 2014) has a keypoint for the top of the head) to the same type of keypoint being associated with different locations on the body. As an example of the latter we find hips annotated in 2D datasets to mismatch the hip joint modeled in the kinematics of the SMPL parameterization – the 2D hip annotations tend to be higher and much wider than they would be in a corresponding SMPL body fit to the same person.

Handling these differences naively by e.g. predicting outputs in the SMPL format while supervising on 2D annotations from COCO and PoseTrack leads to the SMPL meshes being contorted to match the 2D keypoint format. After investigating several different strategies to handle the format mismatch, we settled on a simple but surprisingly effective approach. To better match the SMPL format, we narrow the 2D hip annotations by a constant factor (we use 2 in all of our experiments). In case no ground truth is available for the SMPL β parameters, we regularize them with an L2 penalty to prevent egregious shape fitting. We further regularize predictions through a 2D reprojection error to the face keypoints in COCO. We observe this leads to better estimates of head direction, although precise fitting of head shape could still be improved.