A. Supplementary Materials

In this document, we provide additional details about our method that were not included in the main manuscript, due to space constraints. We include more implementation details about our approach (Sections 2 & 3), we provide more details about the experiments of our paper (Sections 4) and we include the training configurations for Kinetics-400 and AVA for reproducibility.

B. Implementation details

Our complete system for action recognition by tracking integrates multiple sub-systems to combine the recent advancements in 2D Detection, Recognition, 3D Reconstruction, as well as Tracking. We can break the overall pipeline into two parts: **a)** frames-to-entities and **b)** entities-to-action.

The first part is to lift entities from frames (here, we consider entities=people). For this, we use a state-of-the-art tracking algorithm, PHALP [49]. The first step of PHALP is to detect people in each frame using Mask R-CNN [24]. We used Detectron2's [68] new baseline models trained with Simple Copy-Paste Data Augmentation [20] with a RegNet-4gf [47] backbone for the detection task. After detecting people in each frame, PHALP uses HMR [22, 27, 48] to reconstruct each person in 3D. Then, the future location, pose, and appearance of each person are predicted for solving association. PHALP uses Hungarian matching to solve the associations between 3D detections and 3D predictions. Finally, a set of tracks will be returned from the tracking which gives us access to entities (people) over time.

For the second part, which is the main part of our paper, we collect the tracks, generate action labels (either from a pretrained model or from ground truth) and train a transformer model to predict actions from 3D tracks. More details on the network architecture and the training and inference protocol are discussed in the following sections.

B.1. PHALP tracklets

In this work, every person in both Kinetics-400 [29] and AVA [23] is tracked. For this, we used the recently proposed 3D tracking algorithm PHALP [49]. PHALP allows us to track people in the wild very robustly and gives their 3D representations. However, the ground-truth action annotations for AVA are given as bounding boxes at 1 Hz frequency. On a side note, we do not use the ground truth tracking annotations in AVA dataset, which is also only available at 1 Hz. First, we use the PHALP detection model (*e.g.*, Mask R-CNN) to detect humans in the video, whenever a frame does not have ground-truth annotations. If the frame indeed has an annotation, we take the ground-truth bounding boxes as granted and bypass Mask R-CNN detections. Since AVA only has bounding box annotations, and PHALP [49] requires bounding boxes and masks, we use Detectron2 [68] to extract masks from bounding boxes with Mask R-CNN. For the validation set, we used the detections from ACAR [44], which are also only available every 30 frames. Therefore, we used a similar strategy to get tracks from bounding boxes available at 1 Hz. For Kinetics, we run PHALP tracking for the whole sequence, which is typically 10s clips. However, since AVA is much longer than Kinetics (15 min), we run the tracker for 4-second windows, centered around the evaluation frame.

B.2. Architecture details

In all of our experiments, we use a vanilla transformer [58] architecture with 16 layers and width of 512. Each layer has 16 self-attention heads followed by layer-norm [2], and a 2-layer MLP followed by layer-norm. We train all the models with a maximum sequence length of 128 frames per person. In other words, every tracklet is trimmed to have a sequence length of 128 frames. The only data augmentation we use is choosing the starting point of the sequence for random trimming. The transformer blocks are followed by a linear layer that predicts AVA action classes. We train all our models with binary cross-entropy loss.

At training time, we use two types of attention masking. First, since the tracklets are not always continuous due to occlusion and missing detections, we mask the corresponding self-attention of these tokens completely. The loss is not applied to these tokens and this part of the tracklet has no effect on training. The second type of masking is done to simulate these kinds of missing detections at test time. We randomly choose a small number of tokens (based on mask ratio), and replace the person-vector with a learnable mask-token. At the self-attention layer, attention is masked such that these masked-tokens will attend other tokens but other tokens will not attend the masked-tokens. Unlike, the first type of masking, we apply loss on these masked token predictions, since if there is a detection available, then there will be a pseudo-ground truth or ground truth label available for training.

At inference time, we do not do any attention-masking. However, there will be some tracklets with discontinuous detections. At these locations, we use the learned masked-token to infill the predictions for the tracklets. Since we are predicting action labels densely for each frame, we take an average pooling of 12 tokens centered around the annotated detection to minimize the gap between human annotations and model predictions.

B.2.1 Action with 3D Pose

In this subsection, we discuss the network architecture used for recognizing action only with 3D pose information over time. The 3D pose has 226 parameters: 207 ($23 \times 3 \times 3$) parameters for joint angles, 9 for the global orientation of the

Configs	Kinetics-400	AVA
optimizer	AdamW [40]	
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.95$	
weight decay	0.05	
learning rate schedule	cosine decay [41]	
warmup epochs	5	
drop out	0.1	
base learning rate	1e-3	1e-3
layer-wise decay [10]	-	0.9
batch size	64	64
training epochs	30	30
drop path [25]	-	0.1
mask ratio	0.4	0.0

Table 6. **Training Configurations:** We report the training configurations used from training our models on Kinetics-400 and AVA datasets.

person, and 10 for the body shape. In addition to this, the 3D translation of the person in the camera frame is represented by 3 parameters. Overall, in this system, a person-vector has a dimension of 229. This vector is encoded by an MLP with two hidden layers to project this to a 256-dimensional vector. The projected person-vector is then passed to the transformer. We also use the three types of positional encodings for time, track, and space as discussed in the main manuscript (Section 3.1).

B.2.2 Action with 3D Pose and Appearance

To encode a strong contextualized appearance feature, we used MViT [14] pretrained with MaskFeat [64]. The MViT model for AVA takes a sequence of frames and a mid-frame bounding box to predict the action label of the person of interest (this is a classical example of the Eulerian way of predicting action). In this paper, we use an MViT-L 40$\times$3 model that takes a 4-second clip and samples 40 frames with a temporal stride of 3 frames as the input and a bounding box of a person at the mid-frame. This gives a 1152-dimensional feature vector before the linear layer in the MViT classifier. We use this 1152-dimensional feature vector as our contextualized appearance feature and encode it into a 256 dimensional vector by an MLP with two hidden layers. Now, we have a pose vector (256 dim, from the previous section) and an appearance vector (256 dim). We concatenate these two vectors to build our person-vector for 3D pose with appearance, and the final 512-dimensional vector is passed to the transformer.

B.3. Training recipe

As discussed in Section 3 of the main manuscript, we first pretrain our method on Kinetics-400 dataset, using the tracklets obtained from PHALP [49]. Each of these tracklets contains a detection at every frame unless the person is occluded or is not detected due to failure of the detection system. We provide these detection bounding boxes as input to the MViT [14] model and generate pseudo ground truth action labels for the tracklets. Once the labels are generated, we train our model end-to-end, with tracklets as inputs and the action labels as outputs. We use the training configurations in Table 6 for pretraining the model on Kinetics-400 tracklets. Once the model is pretrained on Kinetics tracklets, we fine-tune the model on AVA tracklets (generated by PHALP) with ground truth action labels. Finally, during the fine-tuning stage, we apply layer-wise decay [10] and drop path [25].