Predicting Multi-Person Dynamics

by

Chirag Karia

A thesis submitted to the
School of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of

Master of Science in **Computer Science**

Faculty of Science
University of Ontario Institute of Technology (Ontario Tech University)
Oshawa, Ontario, Canada
December 2023

Thesis Examination Information

Submitted by: **Chirag Karia**

Master of Science in **Computer Science**

Thesis Title: Predicting Multi-Person Dynamics

An oral defense of this thesis took place on December 07, 2023 in front of the following examining committee.

Examining Committee:

Chair of Examining Committee	Dr. Alvaro Quevedo
Research Supervisor	Dr. Faisal Qureshi
Research Co-supervisor	Dr. Kosta Derpanis, York University
Examining Committee Member	Dr. Ken Pu
Thesis Examiner	Dr. Bill Kapralos

The above committee determined that the thesis is acceptable in form and content and that a satisfactory knowledge of the field covered by the thesis was demonstrated by the candidate during an oral examination. A signed copy of the Certificate of Approval is available from the School of Graduate and Postdoctoral Studies.

Abstract

Humans unconsciously model the dynamics of the world around them; for example, we predict the movement of surrounding traffic and pedestrians while driving, or forecast player positions in a game of soccer. Our work builds towards enabling computers with a facet of this ability. Given a video and corresponding bounding box tracks, we propose various methods to predict the future shape, pose, and position of people in unseen frames. Other works that also tackle video-based mesh prediction of humans focus on predicting the shape and pose, ignoring the position of the person in the scene. Additionally, they focus on predicting the future states of each individual in isolation, neglecting how interactions between individuals in a scene can inform their future actions. We present methods to address both of these limitations, and when evaluated on the Human3.6M and 3DPW datasets, we show favorable results to inform future directions of research.

Keywords: human mesh recovery; human mesh prediction; deterministic human motion prediction; multi-person motion prediction

Author's Declaration

I hereby declare that this thesis consists of original work of which I have authored. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners. I authorize the Ontario Tech University to lend this thesis to other institutions or individuals for the purpose of scholarly research. I further authorize the Ontario Tech University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research. I understand that my thesis will be made electronically available to the public.

Chirag Karia

Statement of Contributions

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published. I have used standard referencing practices to acknowledge ideas, research techniques, or other materials that belong to others. Furthermore, I hereby certify that I am sole source of the creative works and/or inventive knowledge described in this thesis.

Acknowledgements

I thank my supervisors, Faisal Qureshi and Kosta Derpanis, for their direction and support in this endeavor. Dr. Qureshi provided me this opportunity, Dr. Derpanis showed me where it could lead. Their advice and enthusiasm was a motivating force from the get-go and they consistently inspired me to meet any complexity with a sense of curiosity and resolve. Alongside my supervisors, I'd also like to thank members of Ontario Tech's Visual Computing Lab for their comradeship, and members of York University's Computational Vision and Imaging Lab for their guidance. Research can rarely be done in silos and both groups have been indispensable in this pursuit.

Next I thank Jonathan Gillet and Wesley Taylor. They took a chance on me by hiring me as a greenhorn and I would not be on this path without them. For this, may your factories ever grow. I also thank my friends, Mohamd Imad and Henry Fung foremost among them, for their encouragement and optimism. In the moments I doubted myself they were there to ground me.

Finally, my deepest thanks to my parents Yogesh and Falguni Karia, without whom none of this would be possible. They have always been my number one supporters and have sacrificed significantly to give me the life I have. I hope that one day I can do for them all they have done for me.

"*It is vital to remember that information – in the sense of raw data – is not knowledge, that knowledge is not wisdom, and that wisdom is not foresight. But information is the first essential step to all of these.*"

– Arthur C. Clarke

Contents

List of Tables

List of Figures

List of Acronyms

3DPW 3D Poses in the Wild.

CNN Convolutional Neural Network.

GAN Generative Adversarial Network.

GELU Gaussian Error Linear Unit.

GPT Generative Pre-trained Transformer.

HMMR Human Mesh and Motion Recovery.

HMR Human Mesh Recovery.

JAT Joint-Aware Transformer.

MAE Mean Absolute Error.

MLP Multi-Layer Perceptron.

MPJAT Multi-Person Joint-Aware Transformer.

MPJPE Mean Per Joint Position Error.

MSE Mean Squared Error.

PA-MPJPE Procrustes Aligned Mean Per Joint Position Error.

PCK Percent Correct Keypoints.

PHD Predicting Human Dynamics.

PHD+T Predicting Human Dynamics and Translation.

SMPL Skinned Multi-Person Linear body model.

TCN Temporal Convolution Network.

Chapter 1

Introduction

With the proliferation of technology that physically operates alongside people, we are seeing a greater need to build machines that can accurately perceive and model human motion. This is something that we take for granted; from a cheetah pouncing on its prey in the final moments of a chase, to a person weaving through a crowd, the ability to model motion and dynamics is a trait that is innate to many living things. If our goal as engineers and scientists is to create robots that can tangibly interact with us, we need to ensure that we instill within them the ability to make sense of us. Be it a self-driving car that shares the streets with pedestrians or a robotic arm working in tandem with an individual on an assembly line, safe operation is contingent on how well these machines can anticipate the actions of the people around them [1, 2].

Generally speaking, there are two parallel streams of research that pertain to modelling human motion. The first stream focuses on parsing video frames and regressing a representation that parameterizes the state of any individuals found. These representations offer different levels of expressiveness and include bounding boxes [3, 4], 2D and 3D keypoints [5, 6], or 3D meshes [7, 8]. The second stream focuses on predicting future human dynamics given a sequence of already parsed data; for example given a stream of past 3D poses, model how the pose changes into the future [9, 10]. Our research lies

at the intersection of these two kindred problems. To this end we introduce a family of methods that aim to first reconstruct the 3D mesh of each individual observed in an input video, and then subsequently predict their future states.

The implications of a method that can reconstruct and then predict 3D representations of individuals in scene given a sequence of frames is many fold. A self-driving vehicle equipped with this ability could predict pedestrian behaviour and use this information in route planning or collision avoidance. In the case of manufacturing, an industrial robot can leverage the motion information of surrounding personnel to engage emergency shutdown procedures before someone comes into harm's way. Finally, in the virtual reality space it can be used to imbibe non-player characters with the ability to preemptively forecast player actions and react in a more life-like manner.

Our primary contribution is an emphasis on leveraging the intrinsic information available in a person's interactions with those around them. For example, a person moving through a crowd will likely travel a route that offers the best balance between a straight path to their destination, and the path of least resistance [11]. Alternatively, when two individuals are coming together for a hand shake, the position of the initiating party's hand can tell us where the hand of the other individual will be in the following moments. Existing works that focus on human mesh prediction are either only interested in modelling scenes with a single person [12, 13] or treat multiple people as an afterthought, predicting the future motion of each individual in isolation [14, 15]. We posit that representations computed in the latter manner are fragmented; by ignoring the interplay between multiple individuals in a scene valuable information is left on the table.

1.1 Contributions

To investigate the concepts described above, we start with an existing single person mesh prediction method, the Predicting Human Dynamics (PHD) model [12] and iteratively

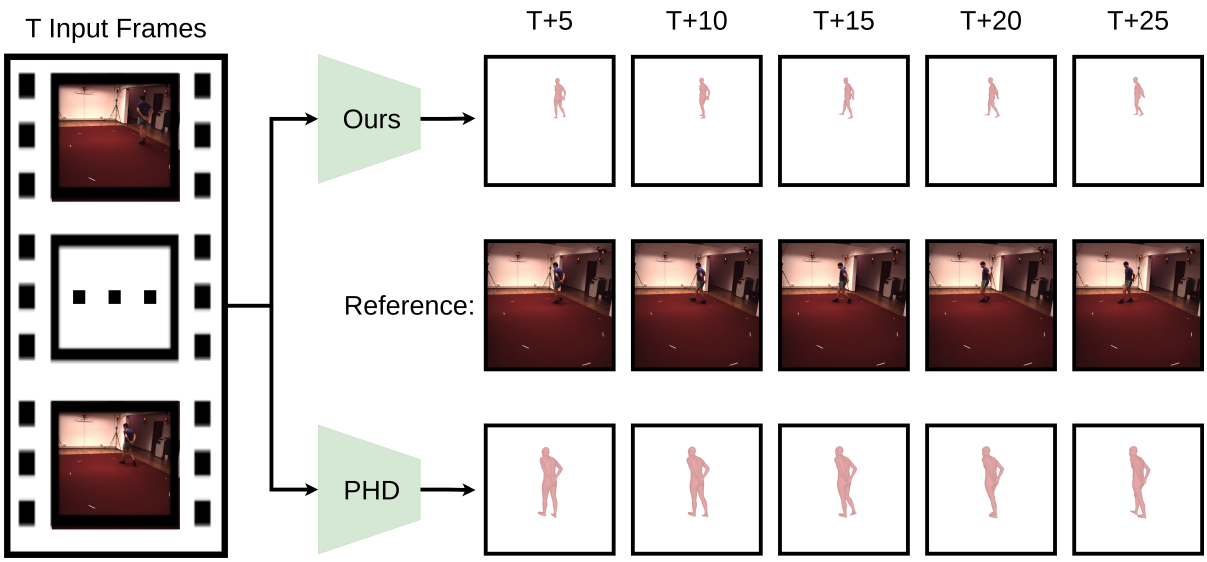


Figure 1.1: **Our Models vs the Original PHD Model.** The existing Predicting Human Dynamics model (bottom) can effectively predict the future shape and pose of an individual given a set of input frames and corresponding bounding boxes; it falls short at estimating the location of an individual in the scene as a whole, instead focusing only on how the limbs move with respect to a static pelvis. The collection of models we propose in this work (top) build on PHD by additionally estimating the translation of an individual, greatly increasing the applicability of these models in the human-robot interaction space.

add to it to formulate a model that can tackle our specific goal of deterministic multi-person mesh prediction. Our contributions include:

1. **Predicting Human Dynamics and Translation (PHD+T):** The original PHD architecture does not model the locomotion of an individual; it assumes the pelvis is fixed in place and focuses only on predicting the shape and pose of future meshes. Our baseline is therefore a modified version of PHD where we also predict the mesh translation, as shown in **Figure 1.1**.

2. **Joint-Aware Transformer (JAT):** As an alternative to the Temporal Convolution Networks [16, 17] that both PHD and PHD+T use, we propose a transformer model that uses GPT-3 [18] style layers augmented with our novel Joint-Aware Bias. The bias allows us inject 3D joint and pose information directly into the computation of the attention weights.

3. **Multi-Person Joint-Aware Transformer (MPJAT):** Our final model extends JAT by contextualizing the reconstructions and predictions of all individuals with respect to each other. We additionally include the interpenetration loss from [19] to explicitly penalize the model for mesh intersection.

1.2 Thesis Outline

The rest of this thesis is organized as follows:

In **Chapter 2** we discuss the current state of human motion modelling based on input RGB video and then cover the research that exists regarding the prediction of future human dynamics given past context.

Chapter 3 provides a background of the constituent elements of our proposed models. Here we introduce the Skinned Multi-Person Linear (SMPL) body model [20], review the HMR head [21] that is responsible for generating mesh parameters from a latent representation, and discuss various neural network architectures for modelling temporal data.

In **Chapter 4** we describe the methods we are proposing. Here we first discuss our modifications to prior work so that the future mesh and trajectory of an individual can be estimated. We then introduce alternative approaches to model the future dynamics of multiple people.

Chapter 5 covers the experiments we ran to evaluate the validity of our contributions. We show quantitative results of our work and also visualize samples from our models.

With **Chapter 6** we review and summarize our work and discuss future directions of research that can stem from our work. Finally we discuss the impact our work, and 3D human mesh reconstruction at large, can have on society as a whole.

Chapter 2

Related Works

In this chapter we outline prior work done towards the two sub-problems we are attempting. We start by looking at methods that regress 3D meshes of individuals in monocular video and then cover research into deterministic and stochastic methods of modelling future motion.

2.1 Reconstructing 3D Human Meshes

A fundamental approach to human mesh reconstruction from video would entail 3D reconstruction of individuals in each frame and then association of individuals across time [22, 23]. The opposite approach would first associate individuals through time, potentially using bounding box based trackers like [3, 4], and then aggregate the information for each individual to produce temporally informed representations [7, 8, 12]. These temporal representations can then be used to construct the 3D meshes of every person where they would be conditioned on relevant data from other frames. Since image based reconstruction is the foundation of the first approach and ideas pertaining to 3D reconstruction from images influence the mesh estimation stage of the latter approach we first provide an overview of image based 3D reconstruction techniques before outlining existing research into 3D mesh reconstruction of humans from monocular video.

2.1.1 Reconstructing 3D Meshes from Images

The first method that could produce a 3D mesh of an individual solely from a single RGB image was SMPLify [24]; it leverages a pretrained 2D joint regressor [25] that estimates 2D keypoints and fits a mesh parameterized using the Skinned Multi-Person Linear (SMPL) body model through an iterative optimization routine. Seminal work in this area is the development of the Human Mesh Recovery (HMR) model [21] by Kanazawa *et al.* Previous methods [26, 27] were bottom-up, and required joint positions be estimated first so that the SMPL model can be fit to them; HMR is a top-down approach that directly estimates SMPL parameters given latent image features and uses an adversarial prior to ensure realistic predictions.

HMR has since become the foundation of many 3D mesh estimation techniques. For example, [28] investigates a synergistic method between HMR and SMPLify, [29] proposes a probabilistic approach for HMR based on latent-flow models, [30, 31] augment HMR with physical constraints like self penetration and floor penetration, and [15] swaps out the CNN backbone with a Vision Transformer [32]. Most relevant to our work are [19, 33, 34, 35] that investigate approaches to reconstruct multiple people. These methods propose techniques that limit collision between individuals and introduce strategies to localize the 3D position of subjects in the scene with respect to the camera.

2.1.2 Reconstructing 3D Meshes from Video

Early work that estimates 3D meshes from monocular video in a markerless manner required multi-camera setups to leverage pixel-wise correspondences [22, 36], however the requirement for multiple cameras makes it difficult to use outside of well formed environments like motion capture studios. One of the first approaches to reconstruct meshes from a single camera video, [23], reconstructs meshes per-frame and then further optimizes both mesh and pose by introducing a bundle adjustment algorithm that smooths

jitter arising from temporal inconsistencies.

The Human Mesh and Motion Recovery (HMMR) model [7] introduced by Kanazawa *et al.* uses Temporal Convolution Networks to contextualize the image features on both past and future frames before using the HMR head to predict the SMPL mesh parameters. VIBE [8] proposes a recurrent model for video mesh estimation with a novel adversarial loss that evaluates for valid transitions between poses, leading to a stronger emphasis on motion dynamics while training. Despite these attempts to leverage temporal information, Choi *et al.* [37] argue that both previous methods are overly dependent on static features corresponding to the specific frame being reconstructed. They blame the residual connection that each of the methods use to merge the temporally aggregated feature with the static feature of the corresponding frame and highlight alternate methods to build temporal representations. More recent research has placed focus on various aspects of the 3D reconstruction pipeline. [38, 39] for example leverage physics simulators and ground plane estimation to model the forces at play in an environment, including gravity, while [40, 41, 42, 43] consider approaches that better account for dynamic cameras and the inevitable occlusions that come about as a result. Other works focus more on downstream applications, [44, 14, 15] are all methods that successively build on each other towards the end goal of a person tracker.

Our work explores how to capture interactions between individuals and use them to construct spatio-temporal representations. Existing research that deals with more than one person generally treats them independently [14, 15, 43], performs a global optimization process after computing separate initial estimates [41, 42], or explicitly negates the spatio-temporal dependencies between individuals [44]. Closest to our own approach is PSVT [45] which also proposes a spatio-temporal framework to reconstruct multiple people from video; while we start with pre-computed bounding box tracks, they disambiguate individuals by using the initial queries of the first decoder layer as slots that their model can assign different people to, similar to object queries in the

Detection Transformer [46]. Another key difference between our approaches is that PSVT splits temporal and spatial conditioning; during attention computation, we allow features corresponding to person A in some frame j to influence the features of another person B in frame t, where $j \leq t$. PSVT on the other hand, constrains temporal conditioning to an individual's own past and performs spatial conditioning only on features from the same frame.

2.2 Modelling Future Human Motion

The prediction of future human motion can generally be separated into deterministic and stochastic methods. Deterministic methods predict a single most likely sequence of events given an input sequence while stochastic methods model distributions, that is they parameterize a multitude of possible futures that can stem from the input. While the work we are proposing in this thesis is deterministic, we outline research on both to provide a comprehensive view of the field. We acknowledge that human behavior is inherently stochastic and deterministic methods of human motion prediction tend to suffer from the regression-to-the-mean problem, where they predict a sequence that averages the many possibilities [47]. We contend however that deterministic approaches are a salient way to test new ideas before developing them to generate predictions stochastically.

2.2.1 Deterministic Methods

Early methods that worked on deterministic prediction of human pose focused on 2D [48], or 3D [49] pose prediction with the use of recurrent models. These methods however are prone to suffering from first-frame discontinuity issues (large jump between last input and first prediction) and error accumulation during inference. Martinez *et al.* [50] show that the first issue can be alleviated with the use of residual connections and the second can be addressed by doing away with teacher forcing [51] or scheduled sampling

[52] by allowing the model to predict autoregressively during training, in the same way it would during inference. To tackle the same issue, Mao *et al.* [53] propose re-evaluating how the input poses are represented. Generally, pose prediction models use joint positions or joint rotations to parameterize an individual's state; the authors instead suggest using a Discrete Cosine Transformation to compute a representation for each joint in the frequency space and then predict future DCT coefficients to model future human pose, an approach that has since been adopted by many recent models [54, 55, 56, 57].

The methods discussed up to this point all focus on modelling the 3D pose of a single individual and assume a static pelvis, ignoring their motion around the scene at large. This formulation however is unsuitable when building models that aim to predict the motion of multiple people; the spatial location of different individuals is inextricably linked to how they interact with those around them [58, 59, 60]. Methods aiming to predict the future pose of multiple people either situate them in world coordinates [10, 61, 62] or use relative coordinates with one individual located at the origin [63]. Closest to our own method is the TBIFormer [10]. While TBIFormer takes 3D pose as an input and not RGB images, they propose a bias on their attention mechanism that is similar in principal to our own. TBIFormer uses Dynamic Time Warping to compute distances between the trajectories of different individuals and maps this to the bias in a non-parametric manner. Our approach (described in **Section 4.3.2**) instead computes the relative joint positions and relative rotations between two poses and lets an MLP learn the mapping to the bias value. The key difference in these approaches is that TBIFormer specifically zeros out any instances in the bias matrix where a query is attending to a key from the same individual, regardless of them being from the same frame or not. We however recognize that there is important spatial and temporal information in the historical motion of a person that can help model their future motion.

Most existing methods that predict future 3D meshes from image data are deterministic as well. The antecedent of our own work is the Predicting Human Dynamics (PHD)

model [12], which takes as input image crops of an individual and predicts their future pose. The key issue with PHD is that similar to previous single-person pose prediction work, it only models a single person with a static pelvis. [14, 15] both extend PHD and propose a prediction methodology that also accounts for movement in a 3D scene. However, the primary purpose of their work is building a person tracking system and they do not evaluate for prediction performance in their experiments. They use prediction as an additional means to ascertain association; each individual is considered in isolation and the focus is on predicting pose one frame into the future, then assigning identity based on similarity of pose and position when that frame is actually captured.

2.2.2 Stochastic Methods

Initial approaches to stochastic human motion forecasting focused on the prediction of trajectories. While there exists much work in this area, most relevant to us are the recent methods that model multi-agent trajectories. [58, 59] leverage a recurrent model with a pooling module that aggregates the hidden-state of an individual with those of their neighbours within a certain distance. They then use it to predict future trajectories; [59] includes a Generative Adversarial Network (GAN) to encourage socially realistic trajectories involving multiple people. [64] drops the distance constraint and adds an attention mechanism [65] over the hidden states of a recurrent model to capture multi-person interactions instead. These approaches however segregate spatial and temporal information where spatial conditioning occurs within independent time steps and temporal conditioning only aggregates the signals of a single individual. AgentFormer [66] introduces a novel attention mechanism, Agent-Aware attention, which imbibes within their transformer the ability to model dependencies between multiple people across space and time simultaneously. This method is the foundation of our own proposed model for capturing multi-person scenes.

Works that explicitly focus on stochastic prediction of human pose [67, 47, 68] learn

a conditional distribution $p(\boldsymbol{X}|\boldsymbol{z}, \boldsymbol{C})$ where $\boldsymbol{X}$ represents a sequence of future poses, $\boldsymbol{z}$ is a latent variable sampled from a Gaussian prior distribution $p(\boldsymbol{z})$, and $\boldsymbol{C}$ is a sequence of past poses. Yuan *et al.* [69] posit that this approach will constrain the diversity of predicted pose sequences $\boldsymbol{X}$ as random sampling from $p(\boldsymbol{z})$ will result in predictions congregating around the modes of the data distribution that are better represented by the training data. To tackle this issue, they propose an inference scheme that involves first sampling a random variable $\boldsymbol{\epsilon}$ from a Gaussian distribution $p(\boldsymbol{\epsilon})$ and then warping it with a set of K mapping functions based on the past poses $\boldsymbol{C}$ to produce a diverse set of K latent variables that can then be used for generation. This idea is leveraged by both [70, 13] to predict future 3D human meshes where [13] is of particular interest to us as it currently appears to be the only work that stochastically predicts future 3D human meshes conditioned on RGB video data.

There has also been much research into combining input modalities to generate more relevant future pose predictions. For example, [71] receives as input SMPL parameters that capture an individual dancing and a corresponding music clip; the model subsequently generates future dance moves that matches the music. Alternatively [72, 73, 74], obtain SMPL parameters and action labels as input where the action labels prime the models' output to enact the given action. Building on the idea of multi-modal inputs, work has been done to predict future human motion with respect to the underlying environment. [9, 75, 76] all are given a scene representation as a 2D image or as a 3D scan; [9, 76] additionally have access to pose history, either as 2D joint positions or as full 3D SMPL parameters. The goal of all three models is to first estimate valid goal locations, plot a valid trajectory to that position, and then finally predict the set of poses that would result in the individual there.

Chapter 3

Technical Preliminaries

3.1 The SMPL Body Model

The Skinned Multi-Person Linear (SMPL) body model [20] is a data driven mapping from a parameterized representation space to a 3D human body mesh. Using parameters that capture the shape and pose of an individual, SMPL can generate a triangular mesh that corresponds to the current state of a body in a fully differentiable manner. The primary goal of the authors of SMPL was to learn a model that can realistically produce a body mesh that can be posed, capture soft-tissue dynamics, and be representative of the spectrum of human body shapes and sizes. Finally, they wanted to ensure it can be used efficiently with existing graphics pipelines.

3.1.1 SMPL Parameters

The SMPL model is parameterized by the pose, shape, and location of the body in the coordinate space. These are referred to as the theta $\boldsymbol{\theta}$, beta $\boldsymbol{\beta}$, and translation $\boldsymbol{\gamma}$ respectively.

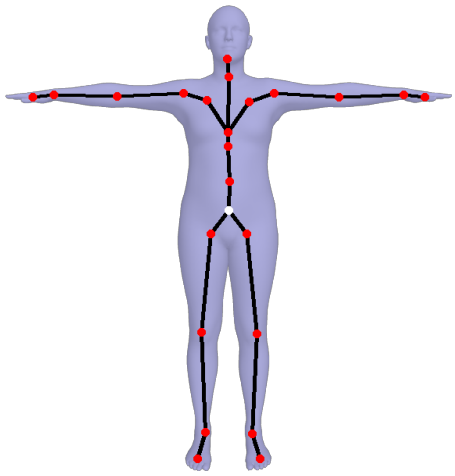


Figure 3.1: **SMPL Kinematic Tree.** The SMPL kinematic tree consists of 24 joints with the pelvis, shown in white, representing the root of the tree. As the root joint, the orientation of the pelvis dictates the global orientation of the entire body with respect to the coordinate frame within which it exists.

Pose Parameters

The theta parameters $\boldsymbol{\theta} \in \mathbb{R}^{216}$ determine the pose of the person and parameterizes the kinematic tree shown in **Figure 3.1**. The first 9 elements of $\boldsymbol{\theta}$ are a 3×3 rotation matrix corresponding to the global orientation of the pelvis. The remaining 207 elements each capture the rotations of 23 joints of the body with respect to a template pose, also as 3×3 rotation matrices.

Shape Parameters

The beta parameters $\boldsymbol{\beta} \in \mathbb{R}^{10}$ represent the overall size and shape of the individual in question. Each of the scalar values of the beta parameters represents an eigenvalue that corresponds to a set of eigenvectors determined through Principal Component Analysis

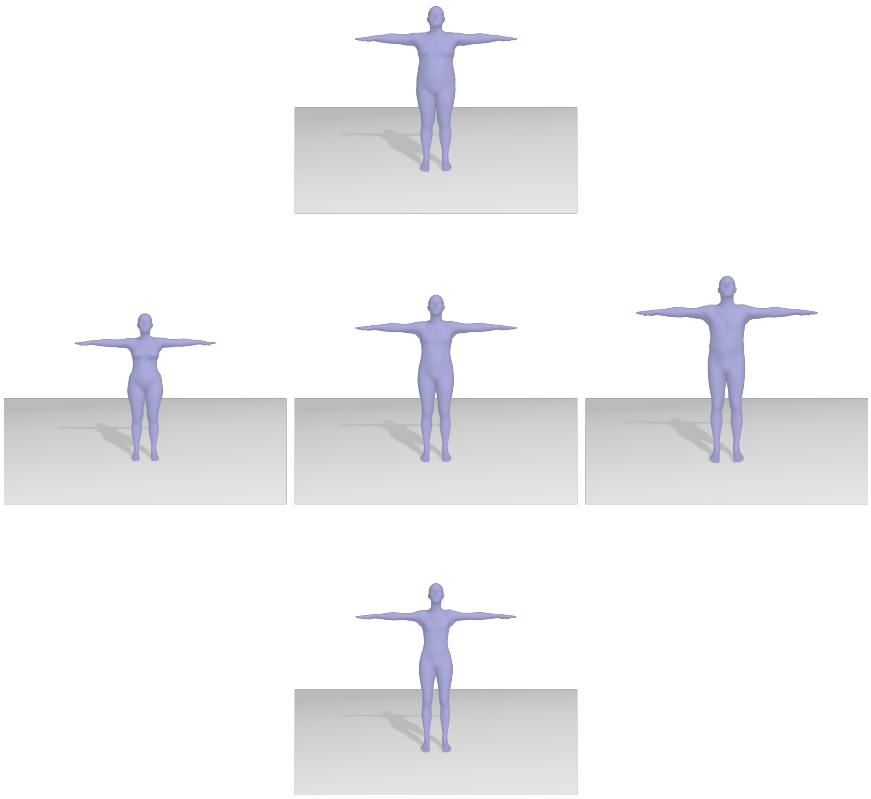


Figure 3.2: **Visualizing the SMPL Shape Parameters.** The center most mesh is a rendering of an SMPL model where all principal components are 0 ($\boldsymbol{\beta} = \mathbf{0}$). The left and right meshes show the effect of setting the first principal component to -2 and +2 respectively. Similarly, the top and bottom meshes show the effect of setting the second principal component to -2 and +2 respectively. It can be observed that the first principal component is primarily influencing the height of the individual, while the second principal component affects the girth of the individual.

and a $\boldsymbol{\beta}$ with all values set to 0 constitutes the shape of the average person. **Figure 3.2** shows the effect of changing the coefficient of the first two principal components which roughly correspond to an individual's height and circumference at the waist.

Translation

The translation $\boldsymbol{\gamma} \in \mathbb{R}^3$ refers to the displacement of the pelvis from the origin in meters. If using world coordinates the translation represents the position of the individual with respect to a predefined location, while in camera coordinates the translation is the

position of the individual with respect to the camera.

3.1.2 Mesh Generation and Joint Estimation

The SMPL model maps the above parameters to a set of vertices $\boldsymbol{V} \in \mathbb{R}^{6890\times 3}$ and we denote the model as the function

$$f_{SMPL} : \{\boldsymbol{\theta},\ \boldsymbol{\beta},\ \boldsymbol{\gamma}\} \mapsto \boldsymbol{V}. \tag{3.1}$$

To compute the 3D spatial positions of joints $\boldsymbol{J} \in \mathbb{R}^{24\times 3}$ we use a pretrained joint regressor such that

$$\boldsymbol{J} = \boldsymbol{W}_J \boldsymbol{V}, \tag{3.2}$$

where $\boldsymbol{W}_J \in \mathbb{R}^{24\times 6890}$. Typically the spatial positions of a set of joints are computed using forward kinematics. Regressing them directly from the mesh vertices however allows for the use of a various collection of joints, making it much easier to use a multitude of datasets where the 3D and 2D joint definitions are not identically defined.

3.2 Human Mesh Recovery

In lieu of directly regressing the vertices of a human mesh it is convenient to instead regress SMPL parameters. With the SMPL model being fully differentiable, gradients from the parameterized mesh can flow backwards to earlier weights during the optimization step [20]. Furthermore by regressing the pose and shape first, adversarial losses and priors can be applied on the pose space without the need for an intermediate step to elicit the pose from methods that directly estimate mesh vertices.

The standard architecture used to estimate SMPL parameters from image data is the HMR head, proposed in End-to-End Recovery of Human Shape and Pose [21]. The HMR

head maps some latent image representation $\boldsymbol{\phi} \in \mathbb{R}^{2048}$ to the mesh pose $\boldsymbol{\theta}$, mesh shape $\boldsymbol{\beta}$, and local camera parameters $\boldsymbol{\pi}$

$$f_{HMR} : \boldsymbol{\phi} \mapsto \{\boldsymbol{\theta},\ \boldsymbol{\beta},\ \boldsymbol{\pi}\}. \tag{3.3}$$

Generally, the latent image features $\boldsymbol{\phi}$ are extracted by cropping the region around a person of interest and then passing the RGB data through 2D convolutional layers. When regressing the pose, HMR opts to use axis-angle representation $\boldsymbol{\theta} \in \mathbb{R}^{72}$ where the first 3 elements represent the global orientation of the pelvis, while the remaining 23×3 elements represent joint rotations. Instead of directly regressing the mesh translation $\boldsymbol{\gamma}$, the HMR head estimates the local camera parameters $\boldsymbol{\pi} \in \mathbb{R}^3 = [s,\ x,\ y]^\top$ which are used in combination with the corresponding bounding box to estimate the translation. The local camera parameters capture the scale s of the mesh relative to the size of the bounding box and 2D x, y offset of the pelvis with respect to the bounding box center.

3.2.1 HMR Head Architecture

The HMR head leverages iterative error feedback [77] to progressively refine predictions. The initial input to the HMR head consists of the image features $\boldsymbol{\phi} \in \mathbb{R}^{2048}$ and the precomputed mean SMPL parameters $\boldsymbol{\Theta}_0 \in \mathbb{R}^{85} = [\bar{\boldsymbol{\theta}}^\top,\ \bar{\boldsymbol{\beta}}^\top,\ \bar{\boldsymbol{\pi}}^\top]^\top$. The mean SMPL parameters $\bar{\boldsymbol{\theta}}$, $\bar{\boldsymbol{\beta}}$ are derived by averaging the respective parameters across the SMPL pseudo-GT of the Human3.6M training set (computed by applying MoSh [78] to motion capture marker data) while the initial local camera parameters $\bar{\boldsymbol{\pi}}$ are set to $s = 0.9$ and $x, y = 0$.

The image features $\boldsymbol{\phi}$ and mean SMPL parameters $\boldsymbol{\Theta}_0$ are concatenated together to form $\boldsymbol{\Omega}_0 \in \mathbb{R}^{2133} = [\boldsymbol{\phi}^\top,\ \boldsymbol{\Theta}_0^\top]^\top$. The HMR head uses $\boldsymbol{\Omega}_0$ to estimate a residual $\Delta\boldsymbol{\Theta}_0$ that is added to $\boldsymbol{\Theta}_0$ to produce $\boldsymbol{\Theta}_1$. Then $\boldsymbol{\Omega}_1 = [\boldsymbol{\phi}^\top,\ \boldsymbol{\Theta}_1^\top]^\top$ is passed through the head again. This process is repeated I times where $\boldsymbol{\Theta}_i = \boldsymbol{\Theta}_{i-1} \oplus \Delta\boldsymbol{\Theta}_{i-1}$ with $\oplus$ denoting elementwise

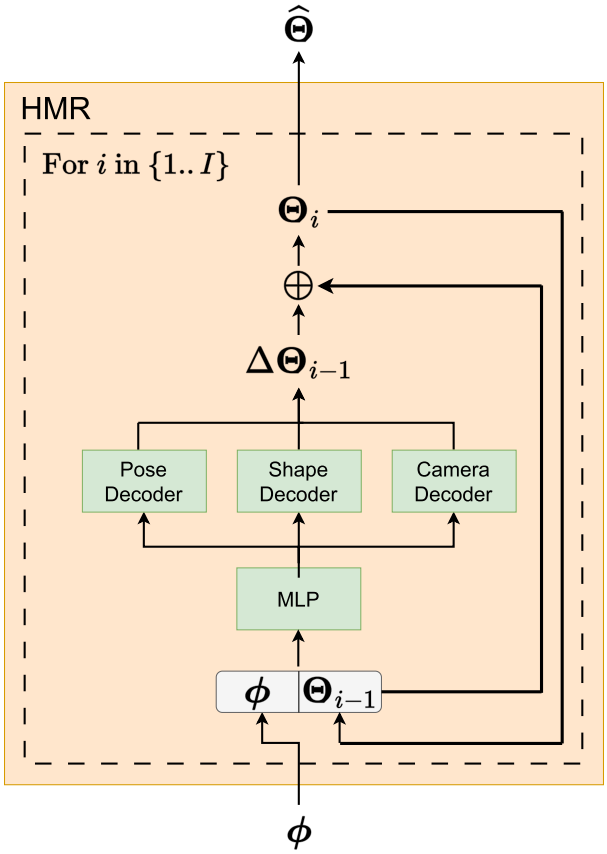


Figure 3.3: **HMR Head Architecture.** Overview of the HMR head. Given an input $\boldsymbol{\phi}$, the HMR head iteratively estimates updates to the initial mean SMPL parameters $\boldsymbol{\Theta}_0$ before outputting the final prediction $\widehat{\boldsymbol{\Theta}}$. There are a total of $I = 3$ iterations, and for each one, the image features are concatenated with the current SMPL estimate $\boldsymbol{\Theta}_i$ to produce $[\boldsymbol{\phi}^\top,\ \boldsymbol{\Theta}_i^\top]^\top$. This concatenated vector is passed through an MLP and then to dedicated decoders to estimate updates for each of the SMPL parameters. The newly updated SMPL parameters are then used to for the next iteration.

addition. The HMR head consists of a Multi-Layer Perceptron and dedicated decoders for $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, and $\boldsymbol{\pi}$ as shown in **Figure 3.3**. The MLP uses two linear layers with weights $\boldsymbol{W}_0 \in \mathbb{R}^{1024\times 2133}$, $\boldsymbol{W}_1 \in \mathbb{R}^{1024\times 1024}$ and the *ReLU* activation function σ to map the input $\boldsymbol{\Omega}_{i-1}$ to $\boldsymbol{z}_{i-1} \in \mathbb{R}^{1024}$ where

$$\boldsymbol{z}_{i-1} = \sigma(\boldsymbol{W}_1\sigma(\boldsymbol{W}_0\boldsymbol{\Omega}_{\boldsymbol{i-1}})). \tag{3.4}$$

The three dedicated decoders are linear layers with weights $\boldsymbol{W}_{\boldsymbol{\theta}} \in \mathbb{R}^{72\times 1024}$, $\boldsymbol{W}_{\boldsymbol{\beta}} \in \mathbb{R}^{10\times 1024}$, $\boldsymbol{W}_{\boldsymbol{\pi}} \in \mathbb{R}^{3\times 1024}$ that take $\boldsymbol{z}_{i-1}$ and predict the corresponding residuals $\Delta\boldsymbol{\theta}_{i-1}$,

$\Delta\boldsymbol{\beta}_{i-1}$, $\Delta\boldsymbol{\pi}_{i-1}$. These are then concatenated together to form $\Delta\boldsymbol{\Theta}_{i-1}$. The final output $\widehat{\boldsymbol{\Theta}}_I$ is computed by adding the residual $\Delta\boldsymbol{\Theta}_{I-1}$ to $\boldsymbol{\Theta}_{I-1}$. At this point rotation matrices for the pose $\boldsymbol{\theta}$ are computed from the axis-angle representation using Rodrigues' rotation formula before being fed to the SMPL model.

3.2.2 Perspective Projection with Local Camera Parameters

To render the predicted mesh on an image, the 3D mesh must be projected onto a 2D image plane. This can be done by using the local camera parameters $\boldsymbol{\pi}$ to perform a weak perspective projection [21, 7, 12] or by using them to estimate the camera-coordinate translation of the mesh and then perform a full perspective projection.

Full Perspective Projection

Jiang *et al.* [19] propose a method to predict the 3D translation $\boldsymbol{\gamma} \in \mathbb{R}^3$ of the mesh in the camera coordinate space given a bounding box, the predicted local camera parameters $\boldsymbol{\pi}$, and ground truth intrinsic camera parameters $\boldsymbol{K}$, where

$$\boldsymbol{K} = \begin{bmatrix} F_x & 0 & \frac{W}{2} \\ 0 & F_y & \frac{H}{2} \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.5}$$

With the estimated $\boldsymbol{\gamma}$, we can project the mesh to the image plane with full perspective projection as described in **Appendix A**. The advantage of a full perspective projection over weak-perspective projection is that by necessity a good estimate of $\boldsymbol{\gamma}$ is required to place the mesh correctly on the 2D image plane [79]. This means that the resulting prediction of $\boldsymbol{\gamma}$ can reasonably be used for down-stream decision making for applications such as human-robot interaction.

To predict the mesh translation $\boldsymbol{\gamma} = [\gamma_x,\ \gamma_y,\ \gamma_z]^\top$ it is first necessary to calculate the depth of the mesh

$$\gamma_z = \frac{2F}{sb}, \tag{3.6}$$

with $b = \max(h, w)$ being the longest edge of a bounding box of height h and width w, the scale parameter s is from $\boldsymbol{\pi}$, and F is the focal length of the dimension corresponding to b from $\boldsymbol{K}$, i.e.,

$$F = \begin{cases} F_y, & \text{if } h > w \\ F_x & \text{otherwise.} \end{cases} \tag{3.7}$$

The x and y components $\boldsymbol{\gamma}_x$, $\boldsymbol{\gamma}_y$ can then be computed using $\boldsymbol{\gamma}_z$ where

$$\begin{aligned} \boldsymbol{\gamma}_x &= \frac{\boldsymbol{\gamma}_z(xw + c_x - \frac{W}{2})}{F}, \text{ and} \\ \boldsymbol{\gamma}_y &= \frac{\boldsymbol{\gamma}_z(yh + c_y - \frac{H}{2})}{F}. \end{aligned} \tag{3.8}$$

Here x, y from $\boldsymbol{\pi}$ are the bounding box offsets, W, H represent the width and height of the full image, and c_x, c_y represent the center of the bounding box with respect to the top left corner of the whole image.

3.2.3 Rotation Representations

The original HMR model [21] and its derivatives [12, 7] estimate the pose and calculate rotation losses in axis-angle form but convert the pose to rotation matrices before passing them to the SMPL model. Other methods have investigated rotation representations more amenable to gradient based learning. HumanShape [27] and UP-3D [26] for example stick to pose estimation using axis-angle, but convert them to rotation matrices when calculating losses, GraphCMR [80] on the other hand estimates the pose by directly regressing the 3×3 rotation matrix for each joint and calculates pose loss with this.

Each of these shows that computing losses on the rotation matrix instead of axis-angle leads to better performance.

Zhou *et al.* [81] empirically show that a 3D rotation matrix $\boldsymbol{M} \in SO(3)$ parameterized with 4 or less dimensions lead to discontinuous representations. In this context discontinuity refers to a many-to-one mapping between a rotation representation (i.e. axis-angle) and its corresponding matrix $\boldsymbol{M}$. Axis-angle rotation defines a rotation of angle r around some axis of rotation $\boldsymbol{v} = [x,\ y,\ z]^\top$. When $\boldsymbol{v}$ is a unit vector $\|\boldsymbol{v}\|_2 = 1$, the axis-angle rotation can be represented as $\boldsymbol{u} = [rx,\ ry,\ rz,]^\top$ where $r = \|\boldsymbol{u}\|_2$ and $\boldsymbol{v} = \frac{\boldsymbol{u}}{r}$. The primary issue with using axis-angle rotations is the discontinuity that occurs at $r = 0$ and $r = 2\pi$. Specifically $\boldsymbol{u} = [0,\ 0,\ 0]^\top$ and any $\boldsymbol{u} = [2\pi x,\ 2\pi y,\ 2\pi z]^\top$ can simultaneously be mapped to the identity rotation $\boldsymbol{I}(3)$.

To tackle this notion of discontinuity Zhou *et al.* [81] propose a continuous 6D representation for rotation. Described further in **Appendix B**, this 6D representation provides a smaller memory footprint with respect to the full 3×3 rotation matrix while also leading to faster convergence of models tasked with regressing rotation [28]. The idea is that the 6D parameterization be used by the network to estimate rotations, then be mapped back to the full rotation matrix when computing losses. Alternatively, estimating the rotation matrix directly can circumvent the need to use the 6D representation but [81] quantitatively shows that this leads to higher error. Following this research, many SMPL based mesh regression methods [14, 19, 8] have adopted the 6D representation with great success.

3.2.4 Losses

The HMR model applies various losses to ensure smooth predictions. They can be broken down into the 3D losses, which influence the model to output predictions that are consistent with the 3D ground truth, a 2D loss which helps guide the prediction of $\boldsymbol{\pi}$ to ensure that the projection to the 2D image plane is satisfactory, and finally an adversarial

loss that ensures that the output meshes are on the manifold of realistic human poses.

3D Losses

There are three different 3D losses that are applied to the HMR model. First there is an MSE loss applied on the estimated joint rotations $\widehat{\boldsymbol{\theta}}$ (in axis-angle or as a rotation matrix), another MSE loss on the estimated shape parameters $\widehat{\boldsymbol{\beta}}$, and a final MSE loss on the estimated 3D joints $\widehat{\boldsymbol{J}}$ extracted from the SMPL mesh with the joint regressor (**Equation 3.2**). All together, the 3D loss is

$$L_{3D} = \lambda_{pose}\|\boldsymbol{\theta} - \widehat{\boldsymbol{\theta}}\|_2^2 + \lambda_{shape}\|\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}\|_2^2 + \lambda_{joints}\|\boldsymbol{J} - \widehat{\boldsymbol{J}}\|_2^2. \tag{3.9}$$

Here $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, and $\boldsymbol{J}$ are the target values and λ_{pose}, λ_{shape}, λ_{joints} are hyperparameters that denote the corresponding loss weights.

2D Loss

The 2D loss is applied to the projected joints, be it via full perspective or weak perspective projection. Assuming f_{proj} is the projection function of choice, L_{2D} is

$$L_{2D} = \lambda_{2D}\|\boldsymbol{j} - f_{proj}(\,\widehat{\boldsymbol{J}}\,)\|_1^1, \tag{3.10}$$

which is the MAE between the ground truth 2D keypoints $\boldsymbol{j}$ and the predicted 3D joints $\widehat{\boldsymbol{J}}$ projected to the image plane.

Adversarial Loss

The final loss the HMR model applies is an adversarial loss, which constrains the mesh to a realistic output space. Without an adversarial loss the model would generate a mesh that minimizes the above defined losses, but with potentially invalid joint rotations [21]. The advantage of a parametric mesh model like SMPL is that a set of discriminators

can be trained directly on its input space, meaning that datasets that provide SMPL parameters can be sampled from to learn the distribution of valid joint rotations and shape parameters.

Inspired by Adversarial Inverse Graphics Networks [82], the authors refer to the set of discriminators as an adversarial prior; the discriminators consist of 23 total joint discriminators that each consider the validity of a single joint (excluding the global orientation/pelvis), a shape discriminator that evaluates the $\boldsymbol{\beta}$ parameters, and a full pose discriminator that looks at the collection of joints as a whole. Each discriminator D_i from the set of 25 discriminators is trained using the least squares GAN loss [83]

$$\min_{D_i} L_{Disc}(D_i) = \mathbb{E}_{\boldsymbol{\Theta} \sim p_{data}} \left[(D_i(\boldsymbol{\Theta}) - 1)^2 \right] + \mathbb{E}_{\boldsymbol{\Theta} \sim p_{\text{gen}}} \left[(D_i(G(\boldsymbol{\phi}))^2 \right], \tag{3.11}$$

where the first term encourages the discriminators to output a value of 1 if a sample is from the data distribution and the second term encourages the discriminators to output a value of 0 if the sample is generated by the HMR model given some image features $\boldsymbol{\phi}$.

The adversarial loss L_{Adv} is applied to the HMR model itself and it quantifies how well the generated samples match the data distribution with

$$L_{Adv} = \lambda_{Adv} \sum_{i=1}^{25} \mathbb{E}_{\boldsymbol{\Theta} \sim p_{\text{gen}}} \left[(D_i(G(\boldsymbol{\phi})) - 1)^2 \right]. \tag{3.12}$$

This influences the HMR model to maximize the belief of each discriminator that the generated sample is from the true data distribution. While training, both the adversarial prior and the HMR model are trained jointly. The gradients from the discriminator loss L_{Disc} are used to update the parameters of the discriminators, while the gradients from the adversarial loss L_{Adv} update the parameters of the HMR model.

3.3 Temporal Convolution Networks

Temporal convolution networks (TCNs) describe a broad approach to sequence modelling using 1D convolution. First introduced by van den Oord *et al.* [16] for generative speech synthesis and then further refined by Lea *et al.* to tackle action segmentation and detection [17], TCNs have been applied with success in the human mesh reconstruction space to regress and smooth human meshes from RGB video [7] and to predict the dynamics of mesh pose from RGB video [12].

3.3.1 1D Causal Convolutions

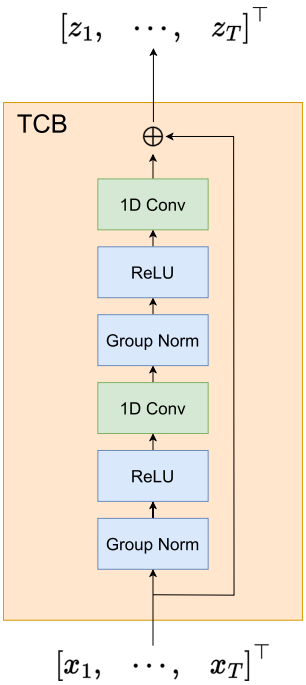
Given a sequence of input vectors $[\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T]^\top \in \mathbb{R}^{T \times c_{in}}$, a 1D causal convolution operation applies 1D convolution such that any output vector $\boldsymbol{z}_t \in \mathbb{R}^{c_{out}}$ is conditioned solely on information up to and including $\boldsymbol{x}_t$. With a filter $\boldsymbol{W} \in \mathbb{R}^{c_{out} \times k \times c_{in}}$ and a bias vector $\boldsymbol{b} \in \mathbb{R}^{c_{out}}$ the response at channel a of an output vector $\boldsymbol{z}_t$ can be calculated as

$$\boldsymbol{z}_{t,a} = \sum_{i=1}^{k} \sum_{j=1}^{c_{in}} \left(\boldsymbol{W}_a \otimes \begin{bmatrix} \boldsymbol{x}_{t-(k-1)}, & \cdots, & \boldsymbol{x}_t \end{bmatrix}^\top \right)_{i,j} + \boldsymbol{b}_a \tag{3.13}$$

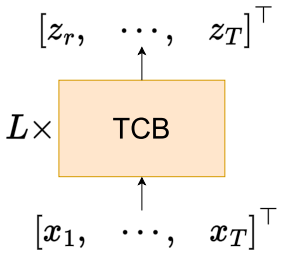
where $[\boldsymbol{x}_{t-(k-1)}, \cdots, \boldsymbol{x}_t]^\top \in \mathbb{R}^{k \times c_{in}}$ is a k length slice from the input sequence and $\otimes$ refers to elementwise multiplication. By top padding the whole input sequence $k-1$ times with the first element $\boldsymbol{x}_1$ before convolution, we can ensure that causality is enforced and the output sequence $[\boldsymbol{z}_1, \cdots, \boldsymbol{z}_T]^\top$ is of the same length T as the input sequence.

3.3.2 Temporal Convolution Network

Temporal convolution blocks are stacks of multiple 1D convolutional layers interleaved with residual connections, normalization layers, and activation functions. Following the architecture in [7], **Figure 3.4a** shows one possible realization of a temporal convolution block. These temporal convolution blocks are in turn stacked multiple times to form a



(a) **Temporal Convolution Block**



(b) **Temporal Convolution Network**

Figure 3.4: **(a)** A temporal convolution block; the inputs to all 1D conv layers are top padded with $k-1$ duplicates of the first element of the sequence. **(b)** A TCN consisting of L blocks. The first $r-1$ elements are dropped from the final output as they are not conditioned on r elements.

Temporal Convolution Network (TCN), where L consecutive temporal convolution blocks induce a total receptive field of

$$r = L(2k-2)+1. \tag{3.14}$$

This results in every output $\boldsymbol{z}_t$ aggregating the information of r total inputs; a temporal convolution network can therefore be treated as the function

$$f_{TCN} : \left[\boldsymbol{x}_{t-(r-1)}, \quad \cdots, \quad \boldsymbol{x}_t\right]^\top \mapsto \boldsymbol{z}_t, \tag{3.15}$$

which summarizes a sequence of r vectors into a single output vector. To ensure that each output $\boldsymbol{z}_t$ of the temporal convolution network is fully conditioned on r inputs, the first $r-1$ outputs of the final temporal convolution block are dropped since any $\boldsymbol{z}_t$ with

$t < r$ does not have access to data before the start of the sequence.

Finally, it is worth noting that while f_{TCN} is defined as a mapping from a sequence of vectors to a single vector, the model can in fact compute the output sequence $[\boldsymbol{z}_r, \cdots, \boldsymbol{z}_T]^\top$ from the full input sequence in one forward pass as shown in **Figure 3.4b**. This concurrency is due to convolution operations being embarrassingly parallel, resulting in each 1D convolution layer being dependent only on the outputs of the preceding layers. In other words, unlike recurrent neural networks there is no requirement to compute the output element by element.

3.4 Transformers

Transformers are the most recent approach to modelling sequence information. First developed to tackle language translation [65], they have since been applied to a wide variety of tasks including image classification [32], object detection [46], and long-term human motion prediction [9]. The key to the transformer architecture is in its use of the attention mechanism. Similar to 1D causal convolution, the attention mechanism provides a means to aggregate sequence information; they differ in the locality of the information they can model. A single 1D convolution layer can only generate outputs that are a function of k consecutive elements (its filter size). To increase the number of elements that an output is conditioned on, you must either increase the filter size, or stack multiple 1D convolution layers. The attention mechanism on the other hand can simultaneously contextualize an entire sequence at once and effectively model the dependencies between the elements of a sequence.

3.4.1 Scaled Dot-Product Attention

Illustrated in **Figure 3.5**, scaled dot-product attention projects the input sequence $\boldsymbol{X} \in \mathbb{R}^{T \times c_{in}} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T]^\top$ to three different representation spaces referred to as the query

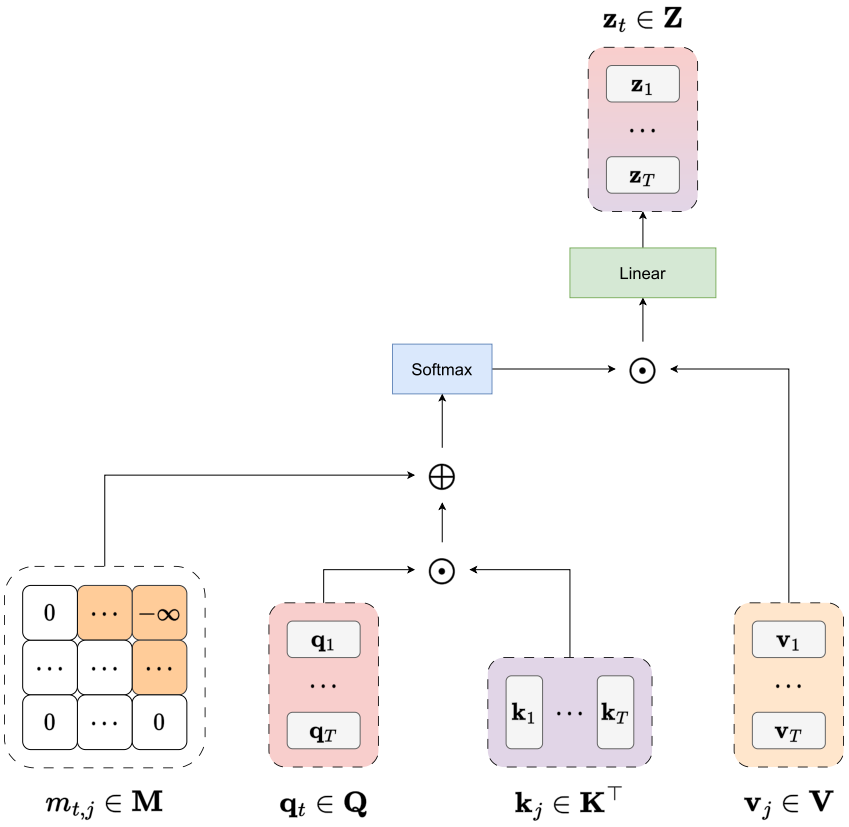


Figure 3.5: **Scaled Dot-Product Attention.** A visual representation of how the attention mechanism uses the $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ matrices to construct the output sequence $\boldsymbol{Z}$. With $\odot$ referring to matrix multiplication and $\oplus$ referring to elementwise addition, the result of softmax$(\boldsymbol{Q}\boldsymbol{K}^\top \oplus \boldsymbol{M})$ is the masked attention matrix $\boldsymbol{\alpha}$. These attention weights can then be used to linearly combine the vectors in $\boldsymbol{V}$ where $\boldsymbol{\alpha}_{t,j}$ prescribes the contribution of $\boldsymbol{v}_j$ when computing $\boldsymbol{z}_t$.

$\boldsymbol{Q} \in \mathbb{R}^{T \times d_k}$, key $\boldsymbol{K} \in \mathbb{R}^{T \times d_k}$, and value $\boldsymbol{V} \in \mathbb{R}^{T \times d_v}$ matrices. These matrices are linear projections of $\boldsymbol{X}$ with

$$\mathbf{Q} = \boldsymbol{X}\boldsymbol{W}_Q, \qquad \mathbf{K} = \boldsymbol{X}\boldsymbol{W}_K, \qquad \mathbf{V} = \boldsymbol{X}\boldsymbol{W}_V. \tag{3.16}$$

Here, the learned weights are $\boldsymbol{W}_Q \in \mathbb{R}^{c_{in} \times d_k}$, $\boldsymbol{W}_K \in \mathbb{R}^{c_{in} \times d_k}$, and $\boldsymbol{W}_V \in \mathbb{R}^{c_{in} \times d_v}$.

Loosely speaking a query vector $\boldsymbol{q}_t$ from $\boldsymbol{Q}$ represents an element of interest t for which we want to generate an output $\boldsymbol{z}_t$, a key vector $\boldsymbol{k}_j$ from $\boldsymbol{K}$ represents the "meta" data used to evaluate how well j responds to the query, and a value vector $\boldsymbol{v}_j$ from $\boldsymbol{V}$

represents the information from j that is most relevant to constructing $\boldsymbol{z}_t$. The essence of attention lies in the attention matrix $\boldsymbol{\alpha} \in \mathbb{R}^{T \times T}$ which encodes pairwise relationships between some $\boldsymbol{q}_t$, $\boldsymbol{k}_j$ and is computed in parallel for all pairs of $\boldsymbol{q}$ and $\boldsymbol{k}$ with

$$\begin{aligned} \boldsymbol{A} &= \frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d_k}}, \\ \boldsymbol{\alpha} &= \text{softmax}\left(\boldsymbol{A} \oplus \boldsymbol{M}\right), \end{aligned} \tag{3.17}$$

where $\boldsymbol{M}$ refers to an optional masking matrix and softmax is applied to each row of the matrix. Ultimately, row t, column j of $\boldsymbol{\alpha}$ captures how well the key $\boldsymbol{k}_j$ matches a query $\boldsymbol{q}_t$. In certain cases it is necessary to prevent a query from attending to all available keys (e.g temporal data) and is accomplished with the masking matrix $\boldsymbol{M}$; to ensure causality for example, any $\boldsymbol{M}_{t,j}$ where $j > t$ is set to $-\infty$. When the masking matrix $\boldsymbol{M}$ is added to $\boldsymbol{A}$ and followed by a softmax, any element in $\boldsymbol{A} \oplus \boldsymbol{M}$ equal to $-\infty$ will become 0 in $\boldsymbol{\alpha}$, as shown in **Figure 3.6**.

The final contextualized output is constructed by taking the matrix product of $\boldsymbol{\alpha}$ and $\boldsymbol{V}$ and using $\boldsymbol{W}^O \in \mathbb{R}^{d_v \times c_{out}}$ to project the result to the output representation space $\boldsymbol{Z} \in \mathbb{R}^{T \times c_{out}} = [\boldsymbol{z}_1, \cdots, \boldsymbol{z}_T]^\top$ with

$$\boldsymbol{Z} = \boldsymbol{\alpha}\boldsymbol{V}\boldsymbol{W}^O. \tag{3.18}$$

3.4.2 The Transformer Model

Encoder-Decoder Transformers

The original transformer model [65] is proposed with an encoder-decoder architecture. As shown in **Figure 3.7** the encoder maps the input sequence to a latent representation

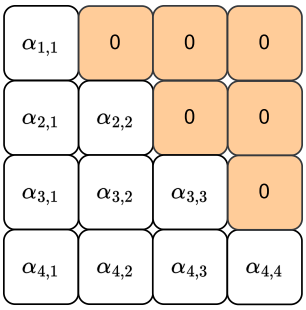


Figure 3.6: **Causally Masked Attention Weights.** Representation of attention weights $\boldsymbol{\alpha}$ after replacing every $\boldsymbol{A}_{t,j}$ where $j > t$ with $-\infty$. After row-wise application of the softmax function every $-\infty$ in $\boldsymbol{A}$ reduces to 0 in $\boldsymbol{\alpha}$ meaning that element $\boldsymbol{v}_j$ has no contribution in the final computation of $\boldsymbol{z}_t$.

using self-attention while the decoder maps the latent representation and decoder inputs to the next output element with cross-attention. Cross-attention is simply an extension of the self-attention mechanism where the query, key, and value matrices are generated from two different representation spaces instead of the same one. In the case of cross-attention in the decoder, the $\boldsymbol{K}$ and $\boldsymbol{V}$ matrices are generated using the latent representation from the encoder and the $\boldsymbol{Q}$ matrix is generated from the input to the decoder block after passing through a masked self-attention layer.

To efficiently train transformer models, teacher forcing is required. With teacher forcing, while a model is being trained the model has access to the entire input sequence, including the elements it is trying to predict. In this case, the transformer receives the target sequence right-shifted as input to the decoder block. Since the model already has access to any $\boldsymbol{x}_j$ with $j > t$ within the decoder's input, future elements must be masked when computing $\widehat{\boldsymbol{x}}_t$.

Decoder Only Transformers

Liu *et al.* [84] introduce the idea of decoder only transformer networks where the encoder and the cross attention layer in the decoder are removed, resulting in there being no distinction between input elements and output elements within the model itself. These models are referred to as decoder only is because the masking process on the self-attention

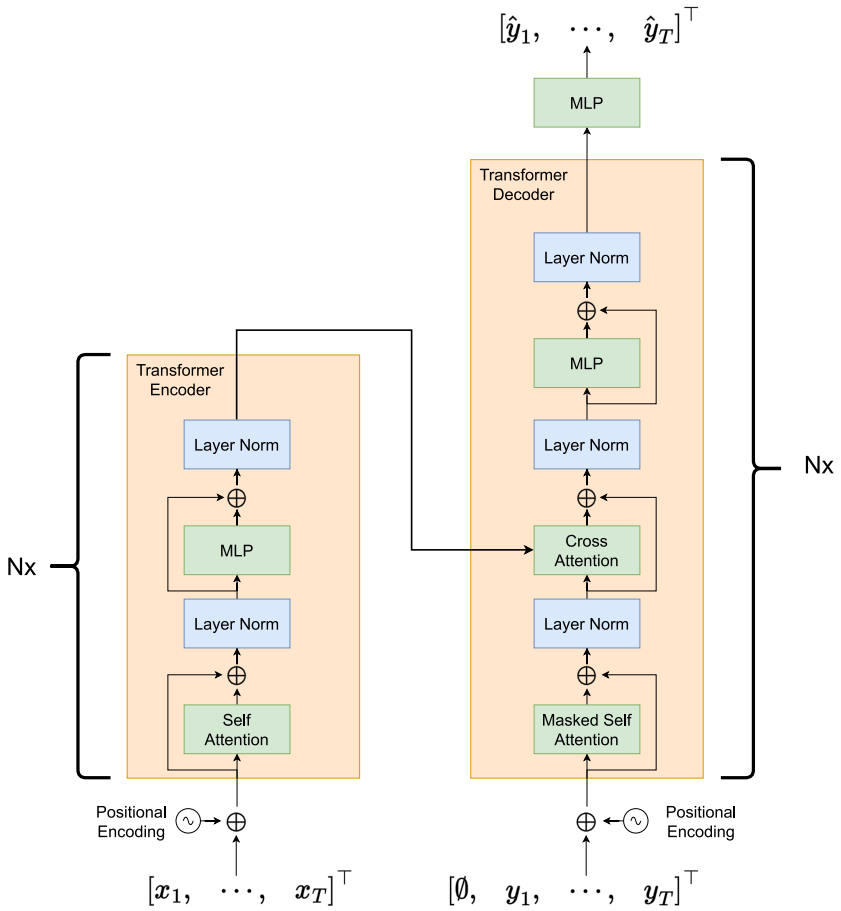


Figure 3.7: **The Transformer Architecture**. The transformer is composed of N stacks of encoder blocks and N stacks of decoders blocks. The output of the final encoder block is used to generate the keys $\boldsymbol{K}$ and values $\boldsymbol{V}$ for the cross attention layer of every decoder block. The queries $\boldsymbol{Q}$ are computed from the input of the decoder block. While training, the input to the first decoder block is the ground truth target sequence, shifted right where $\emptyset$ is a padding element. During inference, the output of the decoder is fed back as its own input and predictions are made autoregressively.

layer of the original transformer decoder is preserved.

Generative Pre-trained Transformers (GPTs) are a collection of large language models [85, 86, 18] developed by OpenAI that also leverage the decoder only framework proposed by Liu *et al.* GPT models can generate long and coherent documents, perform translation, and answer questions [18] given some initial context. What's most interesting is all of these tasks are done using the token prediction paradigm where the model is given an input sequence and simply predicts the subsequent word representations autoregressively.

Where the GPT models differ from Liu *et al.* is that instead of predicting the next

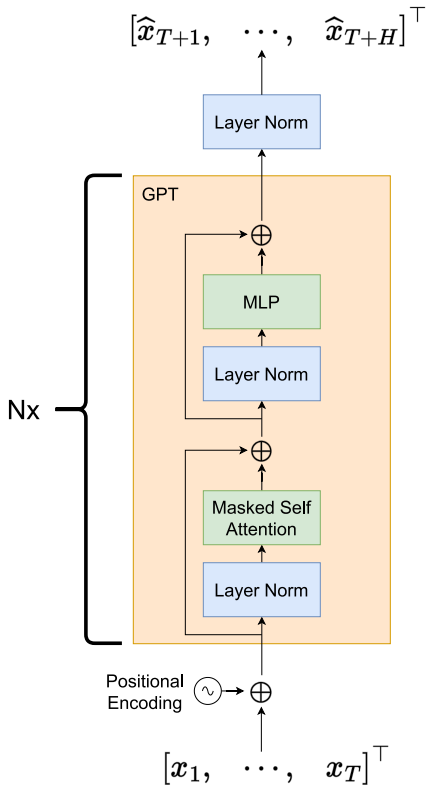


Figure 3.8: **GPT Architecture.** The GPT model drops the transformer encoder and the cross attention layer, instead using just the decoder block. Like the original transformer, GPT is trained using teacher forcing, meaning it receives the entire sequence as input with future elements being masked out in the attention layer. During inference, the GPT model also performs prediction autoregressively with the output being used as input for subsequent steps.

token for every single element in the input, the first element in the output sequence resides after the last element of the input. That is, given some ground truth sequence $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{T+H}]^\top$ the input is $[\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T]^\top$ and the output is $[\widehat{\boldsymbol{x}}_{T+1}, \cdots, \widehat{\boldsymbol{x}}_{T+H}]^\top$. The authors formulate GPT as an auto-regressive network that models the joint probability of a sequence as a product of the conditional probabilities of the past, denoted as

$$p(\boldsymbol{X}) = \prod_{t=T+1}^{T+H} p(\boldsymbol{x}_t | \boldsymbol{x}_1, \cdots, \boldsymbol{x}_{t-1}). \tag{3.19}$$

Positional Encoding

While masking future elements ensures that information does not travel backwards with respect to time, the attention mechanism is inherently permutation invariant [65] and has no information regarding the ordering of unmasked elements in the sequence. To alleviate this, the authors of the original paper suggest adding a positional encoding to embed order into the each element of the sequence. Some possible options for positional encoding include a sinusoidal positional encoding, a learned positional encoding, and a relative positional encoding [87].

Given an input sequence $\boldsymbol{X}$ the positional encoding $\boldsymbol{p}_t$ for element $\boldsymbol{x}_t$ when using sinusoidal embeddings is computed as

$$\begin{aligned} \mathbf{p}_{t,2i} &= \sin\left(\frac{t}{10000^{2i/c_{in}}}\right), \\ \mathbf{p}_{t,2i+1} &= \cos\left(\frac{t}{10000^{2i/c_{in}}}\right) \end{aligned} \tag{3.20}$$

where $i \in \{0, \cdots, \frac{1}{2}c_{in}\}$ and $\boldsymbol{p}_{t,j}$ represents the value at element t, column j of $\boldsymbol{P} \in \mathbb{R}^{T \times c_{in}}$. The input to the transformer model then becomes

$$\boldsymbol{X}' = \boldsymbol{X} \oplus \boldsymbol{P}, \tag{3.21}$$

where $\oplus$ is elementwise addition. Alternatively, instead of adding the positional encoding, some transformer based models [66] suggest passing $\boldsymbol{X}$ through an MLP where $\boldsymbol{P}$ is concatenated to an intermediate representation then projected again, as described by

$$\boldsymbol{X}' = \sigma\left(\left[\sigma(\boldsymbol{X}\boldsymbol{W}_1),\ \boldsymbol{P}\right]\boldsymbol{W}_2\right). \tag{3.22}$$

Here σ refers to the *ReLU* activation function, and the MLP consists of weights $\boldsymbol{W}_1 \in \mathbb{R}^{c_{in} \times c_{in}}$ and $\boldsymbol{W}_2 \in \mathbb{R}^{2c_{in} \times c_{in}}$.

Chapter 4

Methodology

This work investigates the viability of using autoregressive models to generate plausible human motion in the short-term future given some initial video footage. Pointedly, we are interested in evaluating whether contextualizing the interactions between individuals can lead to better results relative to models that condition on people individually. In this section we first formalize the problem we are considering, illustrate a baseline model that can be applied to it, and then introduce our Joint-Aware Transformer and Multi-Person Joint-Aware Transformer.

4.1 Problem Setup

Given an input video and a set of bounding box tracks, the aim is to autoregressively generate the future mesh of each person in the observed frames up to some predefined time horizon. Concretely, the input consists of an RGB video $\boldsymbol{F} \in \mathbb{R}^{T \times H \times W \times 3}$ with $T = 25$ frames and bounding boxes $\boldsymbol{B} \in \mathbb{R}^{T \times N \times 4}$ where N denotes the number of individuals in the video with valid bounding box tracks. The goal is then to model the function

$$\{\boldsymbol{F}, \boldsymbol{B}\} \mapsto \{\boldsymbol{\Theta}^n_{T+1}, \cdots, \boldsymbol{\Theta}^n_{T+H}\}^N_{n=1}, \tag{4.1}$$

with H denoting the horizon for generation and $\mathbf{\Theta}_t^n = \{\boldsymbol{\theta}_t^n,\ \boldsymbol{\beta}_t^n,\ \boldsymbol{\gamma}_t^n\}$ representing the SMPL parameters (**Section 3.1.1**) for each person n in frame t.

4.2 Predicting Human Dynamics and Translation

The model proposed in Predicting 3D Human Dynamics from Video [12] serves as the foundation of our work and thus an extension of it is one of our baselines. The original PHD model does not predict the translation parameter $\boldsymbol{\gamma}$ of future meshes; the focus is on modelling only the future pose and shape of an individual. Nor does it contextualize information between multiple individuals in the same video, instead predicting the future dynamics of each individual separately.

The objective of our work is to develop a model for future shape, pose, and location of multiple people conditioned on past frames. As shown in **Figure 1.1** we extend the PHD model so that it predicts the future mesh translation, but maintain the isolation between each person. We refer to this as the PHD+T model. Additionally, the original PHD model uses a weak-perspective camera model similar to HMR [21]; since we wish to model the human meshes with realistic distances we opt to use the full-perspective camera model. Finally, we use the 6D rotation representation described in **Appendix B** as it has shown to improve performance.

4.2.1 Model Architecture

The overall architecture of our PHD+T model can be seen in **Figure 4.1**. Given an input video $\boldsymbol{F} = \{\boldsymbol{I}_1, \cdots, \boldsymbol{I}_T\}$, for each frame t and individual n we first crop out the regions contained by the bounding boxes $\boldsymbol{B}_t^n$ and resize them to images of size 224×224. We then pass each crop through a pretrained backbone for feature extraction and the final output feature map is average pooled along the spatial dimensions to produce the per-bbox image features vector $\boldsymbol{\phi}_t^n \in \mathbb{R}^{2048}$.

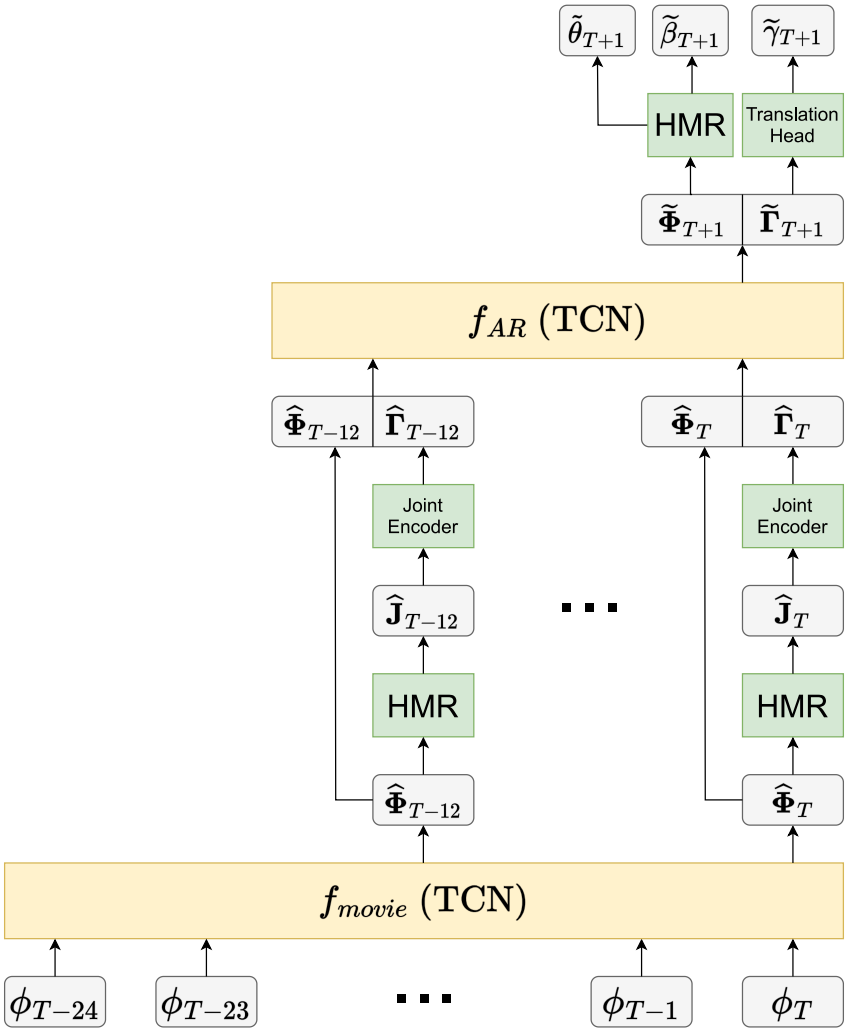


Figure 4.1: **Predicting Human Dynamics and Translation Architecture.** This figure depicts a single forward pass of the PHD+T baseline where both f_{movie} and f_{AR} have a receptive field of $r = 13$; as the model operates on individuals independently, we drop the person index n for compactness. The first TCN f_{movie} aggregates the information from the image features $\boldsymbol{\phi}$ into a set of vectors $\widehat{\mathbf{\Phi}}$ where each vector summarizes information from the 12 preceding frames and the corresponding frame itself. These are concatenated with the encoded joint information $\widehat{\mathbf{\Gamma}}$ and then passed to f_{AR} to produce $\widetilde{\mathbf{\Phi}}_{T+1}$ and $\widetilde{\mathbf{\Gamma}}_{T+1}$ which are used to estimate the future SMPL parameters. Generating multiple time steps into the future is done autoregressively where the output of f_{AR} is fed back to itself as input.

To build a richer latent representation space for modelling dynamics, the image features $\boldsymbol{\phi}_t^n$ for each individual are aggregated using a temporal convolution network f_{movie}. Consisting of three temporal convolution blocks with each 1D convolution layer having a kernel size of $k = 3$, f_{movie} has a total receptive field of $r = 13$. The goal of f_{movie} is to summarize the information from a contiguous set of r frames into a single latent vector $\widehat{\boldsymbol{\Phi}}_t^n \in \mathbb{R}^{2048}$; f_{movie} therefore models the function

$$f_{movie} : \left[\boldsymbol{\phi}_{t-(r-1)}^n, \quad \cdots, \quad \boldsymbol{\phi}_t^n\right]^\top \mapsto \widehat{\boldsymbol{\Phi}}_t^n. \tag{4.2}$$

These temporally aggregated features $\widehat{\boldsymbol{\Phi}}_t^n$ can then be used to estimate the SMPL parameters of individual n in frame t with

$$f_{HMR} : \widehat{\boldsymbol{\Phi}}_t^n \mapsto \widehat{\boldsymbol{\Theta}}_t^n, \tag{4.3}$$

where f_{HMR} follows the HMR head architecture described in **Section 3.2.1**. To compute the translation component $\widehat{\boldsymbol{\gamma}}_t^n$ of $\widehat{\boldsymbol{\Theta}}_t^n$ we leverage the estimated local camera $\widehat{\boldsymbol{\pi}}_t^n$ and input bounding box $\boldsymbol{B}_t^n$ to use the method proposed by Jiang *et al.* [19] as described in **Section 3.2.2**.

The crux of the original PHD model is that once f_{movie} and f_{HMR} can estimate accurate SMPL parameters, the temporally aggregated features $\widehat{\boldsymbol{\Phi}}_t^n$ become a promising representation to use as a basis for prediction of future human meshes. By training an additional TCN f_{AR} to estimate $\widetilde{\boldsymbol{\Phi}}_{t+1}^n$, f_{HMR} can extract the future pose $\widetilde{\boldsymbol{\theta}}_{t+1}^n$ and shape $\widetilde{\boldsymbol{\beta}}_{t+1}^n$. Formulating f_{AR} as a TCN with the same architecture as f_{movie}, we can train it to model

$$f_{AR} : \left[\widehat{\boldsymbol{\Phi}}_{t-(r-1)}^n, \quad \cdots, \quad \widehat{\boldsymbol{\Phi}}_t^n\right]^\top \mapsto \widetilde{\boldsymbol{\Phi}}_{t+1}^n. \tag{4.4}$$

The above approach works if only the future pose and shape are of interest. Estimating the translation $\widehat{\boldsymbol{\gamma}}_t^n$ requires both the estimated local camera parameters $\widehat{\boldsymbol{\pi}}_t^n$ and the

corresponding bounding box $\boldsymbol{B}_t^n$. Any future $\widetilde{\boldsymbol{\gamma}}_j^n$ where $j > T$ cannot be directly extracted from $\widetilde{\boldsymbol{\Phi}}_j^n$ because we do not have access to future bounding box information. We build on the original PHD model by leveraging the 3D joint information $\widehat{\boldsymbol{J}}_t^n \in \mathbb{R}^{24\times 3}$ that can be extracted from the estimated SMPL mesh vertices $\widehat{\boldsymbol{V}}_t^n$ through **Equation 3.2** to predict future translation. First we create a joint encoder

$$f_{JE} : \widehat{\boldsymbol{J}}_t^n \mapsto \widehat{\boldsymbol{\Gamma}}_t^n, \tag{4.5}$$

which maps the joints $\widehat{\boldsymbol{J}}_t^n$ to a latent representation $\widehat{\boldsymbol{\Gamma}}_t^n \in \mathbb{R}^{128}$. Then f_{AR} is modified to model future $\widetilde{\boldsymbol{\Phi}}_{t+1}^n$ and $\widetilde{\boldsymbol{\Gamma}}_{t+1}^n$ concatenated together; i.e.,

$$f_{AR} : \left[\begin{array}{c} \widehat{\boldsymbol{\Phi}}_{t-(r-1)}^n \\ \hdashline \widehat{\boldsymbol{\Gamma}}_{t-(r-1)}^n \end{array}, \cdots, \begin{array}{c} \widehat{\boldsymbol{\Phi}}_{t}^n \\ \hdashline \widehat{\boldsymbol{\Gamma}}_{t}^n \end{array}\right]^\top \mapsto \left[\begin{array}{c} \widetilde{\boldsymbol{\Phi}}_{t+1}^n \\ \hdashline \widetilde{\boldsymbol{\Gamma}}_{t+1}^n \end{array}\right]. \tag{4.6}$$

To predict the translation itself, we use a translation head f_{transl} which maps

$$f_{transl} : \widetilde{\boldsymbol{\Gamma}}_{t+1}^n \mapsto \Delta\widetilde{\boldsymbol{\gamma}}_{t+1}^n, \tag{4.7}$$

where $\Delta\widetilde{\boldsymbol{\gamma}}_{t+1}^n$ is used to estimate $\widetilde{\boldsymbol{\gamma}}_{t+1}^n$ with

$$\widetilde{\boldsymbol{\gamma}}_{t+1}^n = \widetilde{\boldsymbol{\gamma}}_t^n + \Delta\widetilde{\boldsymbol{\gamma}}_{t+1}^n. \tag{4.8}$$

This formulation then allows us to predict all necessary SMPL parameters to generate future meshes.

To generate meshes further than one frame into the future, the model is run autoregressively. To estimate the parameters for $\widetilde{\boldsymbol{\Theta}}_{T+2}^n$ for example, we first extract the joint embedding for $\widetilde{\boldsymbol{\Gamma}}_{T+1}^n$ from the SMPL parameters for the previous time step $\widetilde{\boldsymbol{\Theta}}_{T+1}^n$. The input to f_{AR} then consists of the joint embedding sequence $[\widehat{\boldsymbol{\Gamma}}_{T-(r-1)+1}^n, \cdots, \widetilde{\boldsymbol{\Gamma}}_{T+1}^n]^\top$ and the temporal sequence $[\widehat{\boldsymbol{\Phi}}_{T-(r-1)+1}^n, \cdots, \widetilde{\boldsymbol{\Phi}}_{T+1}^n]^\top$. The output of f_{AR} is the prediction of the next $\widetilde{\boldsymbol{\Phi}}_{T+2}^n$ and $\widetilde{\boldsymbol{\Gamma}}_{T+2}^n$ which are used as described previously to estimate the corre-

sponding SMPL parameters $\widetilde{\boldsymbol{\Theta}}^n_{T+2}$. Therefore f_{AR} is used a total of i times to generate mesh parameters $T+i$ steps into the future where each forward pass is increasingly conditioned on the previous outputs of f_{AR} itself.

4.2.2 Training and Model Losses

Training our proposed PHD+T baseline follows a two-step training procedure similar to the original PHD model [12]. First f_{movie} and f_{HMR} are trained together to estimate the SMPL parameters of input frames. We feed f_{movie} all bounding boxes for every frame from $\boldsymbol{I}_1$ to $\boldsymbol{I}_{T+H}$ and get $\widehat{\boldsymbol{\Phi}}^n_r$ to $\widehat{\boldsymbol{\Phi}}^n_{T+H}$. The output head, f_{HMR} then estimates the meshes for every individual in frame $\boldsymbol{I}_r$ to frame $\boldsymbol{I}_{T+H}$ using the estimated temporal features. Once the results converge, we freeze the weights of f_{movie} & f_{HMR} and subsequently train f_{AR} to predict the future meshes in frames I_{T+1} to I_{T+H} using the temporal features from frames $\boldsymbol{I}_r$ to $\boldsymbol{I}_T$.

Following the original PHD model [12] this model leverages the losses used by the HMR model [21], as well as losses more suited to optimizing over temporal data. As we consider each individual in the scene independently, we drop the person index n when describing the losses. The total loss applied to the meshes that are computed using f_{movie} is

$$L_{movie} = \sum_{t=r}^{T+H} \left[L_t + \sum_{\Delta t} L_{t+\Delta t} \right] + \lambda_{const} \sum_{t=r}^{T+(H-1)} \|\widehat{\boldsymbol{\beta}}_t - \widehat{\boldsymbol{\beta}}_{t+1}\|_2^2. \tag{4.9}$$

Here L_t corresponds to

$$L_t = L_{3D} + L_{2D} + L_{Adv} + L_{reg}, \tag{4.10}$$

which consists of the losses described in **Section 3.2.4** with the addition of a shape regularizer L_{reg}. Since the shape parameter of the SMPL model is zero mean by design, the shape regularizer helps to ensure that values don't stray too far from the shape of

the average person with

$$L_{reg} = \lambda_{reg} \|\widehat{\boldsymbol{\beta}}_t\|_2^2. \tag{4.11}$$

The $\sum_{\Delta t} L_{t+\Delta t}$ term in **Equation 4.9** encourages the model to embed temporal information by using $\widehat{\boldsymbol{\Phi}}_t^n$ to estimate the mesh parameters Δt frames into the past where $\Delta t \in \{-5, -10\}$. This is accomplished with additional HMR heads (referred to as the delta predictors) that are tasked with estimating mesh parameters specific to a corresponding offset from frame t; a given $\widehat{\boldsymbol{\Phi}}_t^n$ must therefore embed relevant information to reconstruct the mesh in frame t itself along with frames $t-5$ and $t-10$ where the loss for estimated meshes from all three frames are computed using **Equation 4.10**. The final term in **Equation 4.9** is the shape consistency loss that ensures the predicted shape parameters $\widehat{\boldsymbol{\beta}}$ are similar across frames by directly penalizing the model for the distance between two consecutive shape predictions.

To train f_{AR}, we use

$$\begin{aligned} L_{AR} = \sum_{t=T+1}^{T+H} \Big[L_t + \lambda_{temp} \|\widetilde{\boldsymbol{\Phi}}_t - \widehat{\boldsymbol{\Phi}}_t\|_2^2 + \lambda_{transl} \|\widetilde{\boldsymbol{\gamma}}_t - \widehat{\boldsymbol{\gamma}}_t\|_2^2 \Big] \\ + \lambda_{const} \sum_{t=T+1}^{T+(H-1)} \|\widetilde{\boldsymbol{\beta}}_t - \widetilde{\boldsymbol{\beta}}_{t+1}\|_2^2, \end{aligned} \tag{4.12}$$

which is similar to the original f_{AR} loss proposed by the PHD model with the primary difference being the introduction of the $\|\widetilde{\boldsymbol{\gamma}}_t - \widehat{\boldsymbol{\gamma}}_t\|_2^2$ term. Here, L_t will guide the meshes estimated through f_{AR} towards the groundtruth SMPL parameters, $\|\widetilde{\boldsymbol{\Phi}}_t - \widehat{\boldsymbol{\Phi}}_t\|_2^2$ encourages f_{AR} to predict a temporal representation similar to f_{movie}, $\|\widetilde{\boldsymbol{\gamma}}_t - \widehat{\boldsymbol{\gamma}}_t\|_2^2$ exists to guide the model's estimates of $\widetilde{\boldsymbol{\gamma}}_t^n$, and the last term is the same shape consistency loss from L_{movie}.

4.3 Joint-Aware Transformer

Our next model is an autoregressive transformer that makes use of the GPT-3 [18] architecture for both f_{movie} and f_{AR}. While this model also operates on each individual independently, it contrasts with PHD+T by leveraging additional 3D information made available by f_{movie} to inform our future predictions in the form of a Joint-Aware Bias. This proposed modification to our transformer's self-attention layer serves as an inductive bias by injecting egocentric information into the attention weights used to calculate the outputs.

4.3.1 Model Architecture

The architecture for JAT can be seen in **Figure 4.2**. It has analogous f_{movie} and f_{AR} layers that are responsible for conditioning on input frames and predicting future meshes respectively. One of the key limitations of PHD+T is the receptive field size of $r = 13$ for f_{AR}. A Temporal Convolution Network based f_{movie} must embed the image features $[\boldsymbol{\phi}_1^n, \cdots, \boldsymbol{\phi}_{25}^n]^\top \in \mathbb{R}^{T\times 2048}$ into the temporal representations $[\widehat{\boldsymbol{\Phi}}_{13}^n, \cdots, \widehat{\boldsymbol{\Phi}}_{25}^n]^\top \in \mathbb{R}^{r\times 2048}$ to ensure information from all frames can be communicated to f_{AR}. In a TCN the receptive field is a function of the number of 1D convolution layers and their corresponding kernel sizes, with transformers however the receptive field is based on predefined hyperparameters that limit how far back the attention mechanism can peer and is not inherently determined by architecture. We take advantage of this by training our transformer based f_{AR} so that it can attend to every frame in the input sequence to inform its predictions of future meshes.

The general process is the same as PHD+T where the image features $\boldsymbol{\phi}_t^n$ are computed by extracting bounding box tracks for each individual in the input sequence then passing the crops through a pretrained backbone and an average pooling layer. These image features $\boldsymbol{\phi}_t^n \in \mathbb{R}^{2048}$ are concatenated with normalized bounding box features $\boldsymbol{B}_t^n \in \mathbb{R}^4$

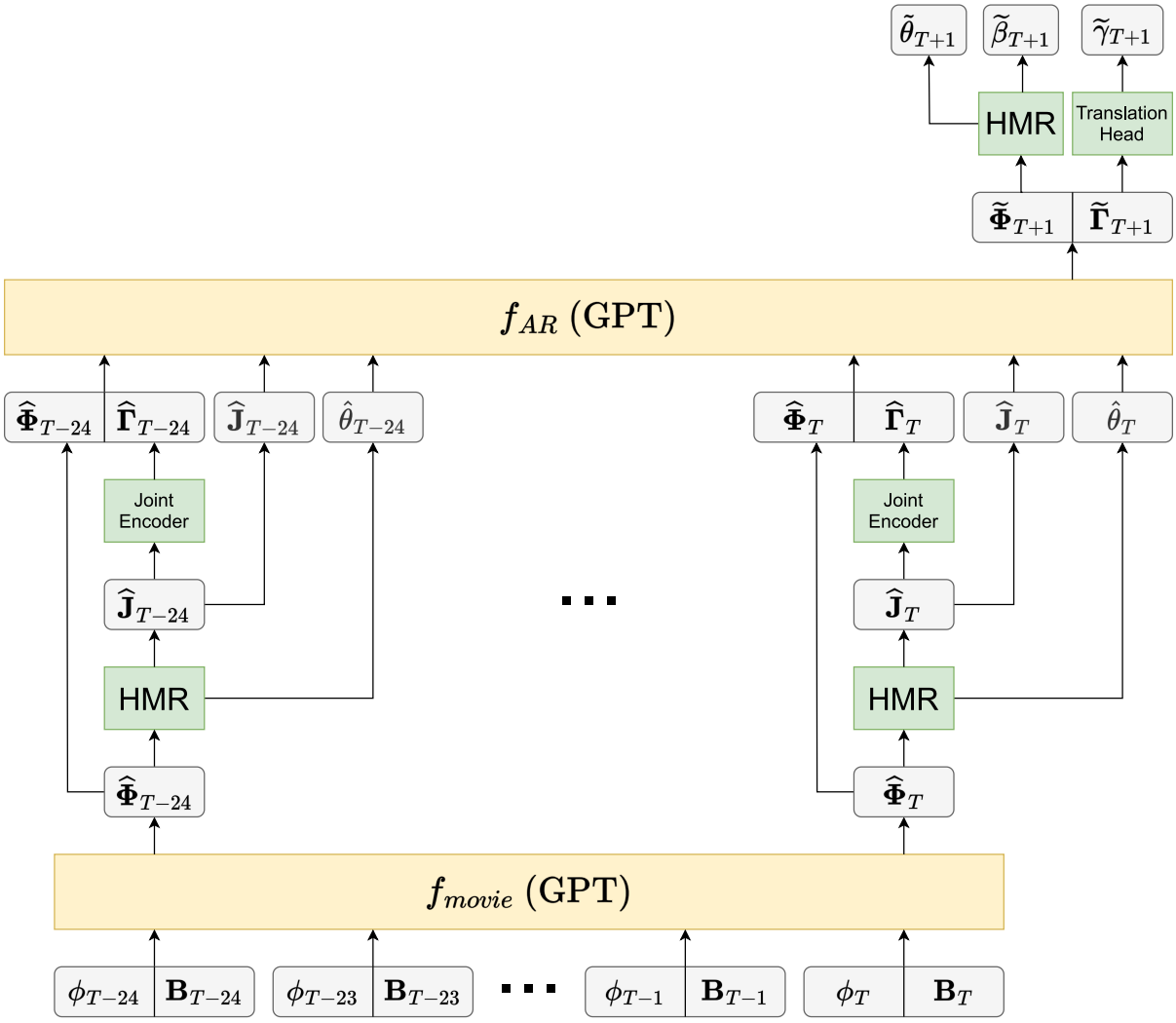


Figure 4.2: **The Joint-Aware Transformer Architecture.** Instead of using TCN layers, JAT uses GPT-style layers, allowing us to make use of our dynamic Joint-Aware Bias to embed the relationships between the joints and poses of different frames directly into the attention weights of f_{AR}. The model first passes the image features $\boldsymbol{\phi}$ and bounding boxes $\boldsymbol{B}$ of the input frames to f_{movie} which contextualizes information from all frames into the temporal features $\widehat{\mathbf{\Phi}}$. We can then use f_{HMR} to estimate the mesh $\widehat{\mathbf{\Theta}}$ from which we extract the joints $\widehat{\boldsymbol{J}}$ and poses $\widehat{\boldsymbol{\theta}}$. The joint encoder produces the latent joint features $\widehat{\mathbf{\Gamma}}$ which is concatenated with the temporal features $\widehat{\mathbf{\Phi}}$ and passed to f_{AR}. Simultaneously, the canonicalized joints and poses are also passed to f_{AR} so that the Joint-Aware Bias encoder can estimate the bias that will be applied to the attention weights of the AR head. Finally, the output of AR head can be used to predict the future $\widetilde{\mathbf{\Theta}}$ where f_{AR} is applied autoregressively to predict meshes more than one frame into the future.

to form our input sequence $\boldsymbol{X}^n \in \mathbb{R}^{t \times 2052}$ where

$$\boldsymbol{X}^n = \left[\begin{array}{c:c:c} \boldsymbol{\phi}_1^n & \cdots & \boldsymbol{\phi}_t^n \\ \hdashline \boldsymbol{B}_1^n & & \boldsymbol{B}_t^n \end{array} \right]^\top . \tag{4.13}$$

This sequence of vectors is passed to f_{movie} which learns the causal mapping

$$f_{movie} : \boldsymbol{X}^n \mapsto \widehat{\boldsymbol{\Phi}}_t^n, \tag{4.14}$$

with f_{movie} parameterized as a GPT layer and $\widehat{\boldsymbol{\Phi}}_t^n \in \mathbb{R}^{512}$ being the temporal representation used downstream by f_{HMR} to estimate the SMPL parameters $\widehat{\boldsymbol{\Theta}}_t^n$. To simultaneously decrease the dimensionality and embed the positional encoding, the input to f_{movie} is first passed through an input encoder which computes the sequence $\boldsymbol{E}^n \in \mathbb{R}^{t \times 512}$ with

$$\boldsymbol{E}^n = \sigma\left([\sigma(\boldsymbol{X}^n \boldsymbol{W}_1),\ \boldsymbol{P}]\, \boldsymbol{W}_2 \right). \tag{4.15}$$

The encoder consists of weights $\boldsymbol{W}_1 \in \mathbb{R}^{2052 \times 512}$ and $\boldsymbol{W}_2 \in \mathbb{R}^{1024 \times 512}$, $\boldsymbol{P} \in \mathbb{R}^{t \times 512}$ is the positional embedding calculated using **Equation 3.20**, and σ refers to the Gaussian Error Linear Unit (GELU) activation function [88]. The encoded sequence is passed to the transformer, which consists of a single GPT-layer (**Figure 3.8**) with one attention head, and yields a causal representation $\widehat{\boldsymbol{\Phi}}_t^n$ for every frame t conditioned on all $\boldsymbol{E}_j^n$ with $j \leq t$. Since we compute the temporal representation $\widehat{\boldsymbol{\Phi}}_t^n$ for all frames in a single forward pass of f_{movie}, we ensure causality by applying a mask $\boldsymbol{M} \in \mathbb{R}^{t \times t}$ to the attention mechanism of f_{movie} as shown in **Figure 3.5**. To predict future SMPL parameters $\widetilde{\boldsymbol{\Theta}}_{t+1}^n$ we use a GPT layer again, this time however we augment it with a dynamic bias which we describe next.

4.3.2 Predicting the Future with a Joint-Aware Bias

When predicting future SMPL parameters $\widetilde{\mathbf{\Theta}}^n_{t+1}$, it is essential to consider the relationships between different parts of the body across time. For example, the position of one joint in earlier frames will influence the position of neighboring joints in subsequent frames. While the attention mechanism in a transformer does this innately, it lacks an inductive bias that could aid with modelling information in 3D space. To this end we augment the attention mechanism of f_{AR} with our proposed Joint-Aware Bias.

We draw inspiration from the approach proposed in [89] by using canonicalized joint and pose features to estimate a bias term used to compute the attention map $\boldsymbol{\alpha}$. Canonicalization in this context refers to transforming the joints and poses to a similar reference frame; in our case we adopt an egocentric reference frame where the global pose and location of the individual in the latest frame (the query) forms the root of the reference frame, a natural extension to works in the 2D trajectory prediction space that leverage varying degrees of egocentric representations [66, 90, 64].

The primary input to the f_{AR} layer of the JAT model is

$$\widehat{\boldsymbol{X}}^n = \begin{bmatrix} \widehat{\mathbf{\Phi}}^n_w \\ \hdashline \widehat{\mathbf{\Gamma}}^n_w \end{bmatrix}, \quad \cdots, \quad \begin{bmatrix} \widehat{\mathbf{\Phi}}^n_t \\ \hdashline \widehat{\mathbf{\Gamma}}^n_t \end{bmatrix}^\top, \tag{4.16}$$

with $\widehat{\boldsymbol{X}}^n \in \mathbb{R}^{(t-w+1)\times 640}$. Here t refers to the query frame and w is the first index of the input, computed using the predetermined receptive field $r = 46$ of f_{AR} where

$$w = \begin{cases} 1 & \text{if } t \leq r, \\ t - r + 1 & \text{otherwise.} \end{cases} \tag{4.17}$$

The matrix $\widehat{\boldsymbol{X}}^n$ is principally the same as the input to the f_{AR} of the PHD+T model; the latent joint representation $\widehat{\mathbf{\Gamma}}^n_t \in \mathbb{R}^{128}$ is computed with the use of a joint encoder as described in **Equation 4.5**, and the main difference is that the transformer based model

uses a smaller temporal representation space where $\widehat{\boldsymbol{\Phi}}_t^n \in \mathbb{R}^{512}$. We additionally pass to f_{AR} a sequence of estimated joints $\widehat{\boldsymbol{J}}^n \in \mathbb{R}^{(t-w+1)\times 24\times 3}$ with $\widehat{\boldsymbol{J}}^n = \left\{\widehat{\boldsymbol{J}}_w^n,\ \cdots,\ \widehat{\boldsymbol{J}}_t^n\right\}$ and estimated poses $\widehat{\boldsymbol{\theta}}^n \in \mathbb{R}^{(t-w+1)\times 24\times 3\times 3}$ where $\widehat{\boldsymbol{\theta}}^n = \left\{\widehat{\boldsymbol{\theta}}_w^n,\ \cdots,\ \widehat{\boldsymbol{\theta}}_t^n\right\}$. Ultimately, f_{AR} models the function

$$f_{AR} : \left\{\widehat{\boldsymbol{X}}^n,\ \ \widehat{\boldsymbol{J}}^n,\ \ \widehat{\boldsymbol{\theta}}^n\right\} \mapsto \left[\begin{array}{c} \widetilde{\widehat{\boldsymbol{\Phi}}}_{t+1}^n \\ \hdashline \widetilde{\boldsymbol{\Gamma}}_{t+1}^n \end{array}\right]. \tag{4.18}$$

The matrix $\widehat{\boldsymbol{X}}^n$ is the main input for the transformer of f_{AR} and thus its downstream representation is used to compute the query, key, and value matrices within the attention mechanism. To this end we first use an input encoder to produce our positionally encoded inputs $\widehat{\boldsymbol{E}}^n \in \mathbb{R}^{(t-w+1)\times 640}$ with

$$\widehat{\boldsymbol{E}}^n = \sigma\left(\left[\sigma(\widehat{\boldsymbol{X}}^n\boldsymbol{W}_1),\ \boldsymbol{P}\right]\boldsymbol{W}_2\right). \tag{4.19}$$

Here the encoder consists of weights $\boldsymbol{W}_1 \in \mathbb{R}^{640\times 640}$, $\boldsymbol{W}_2 \in \mathbb{R}^{1280\times 640}$, the GELU activation function σ, and the positional encoding matrix itself $\boldsymbol{P} \in \mathbb{R}^{(t-w+1)\times 640}$. After the input encoder produces $\widehat{\boldsymbol{E}}^n$, it undergoes layer normalization and is passed to the attention mechanism of f_{AR}.

As the goal is to predict the future temporal representation $\widetilde{\boldsymbol{\Phi}}_{t+1}^n$ and future latent joint representation $\widetilde{\boldsymbol{\Gamma}}_{t+1}^n$, our query matrix $\boldsymbol{Q}$ is computed using only the information pertaining to the most recent time step t. That is, given $\widehat{\boldsymbol{E}}_t^n \in \mathbb{R}^{1\times 640}$ and $\boldsymbol{W}_Q \in \mathbb{R}^{640\times 640}$, we calculate $\boldsymbol{Q} \in \mathbb{R}^{1\times 640}$ with

$$\boldsymbol{Q} = \widehat{\boldsymbol{E}}_t^n\boldsymbol{W}_Q. \tag{4.20}$$

The keys $\boldsymbol{K} \in \mathbb{R}^{(t-w+1)\times 640}$ and values $\boldsymbol{V} \in \mathbb{R}^{(t-w+1)\times 640}$ on the other hand are computed using all available elements in $\widehat{\boldsymbol{E}}^n$. The attention matrix $\boldsymbol{\alpha}$ is then computed with

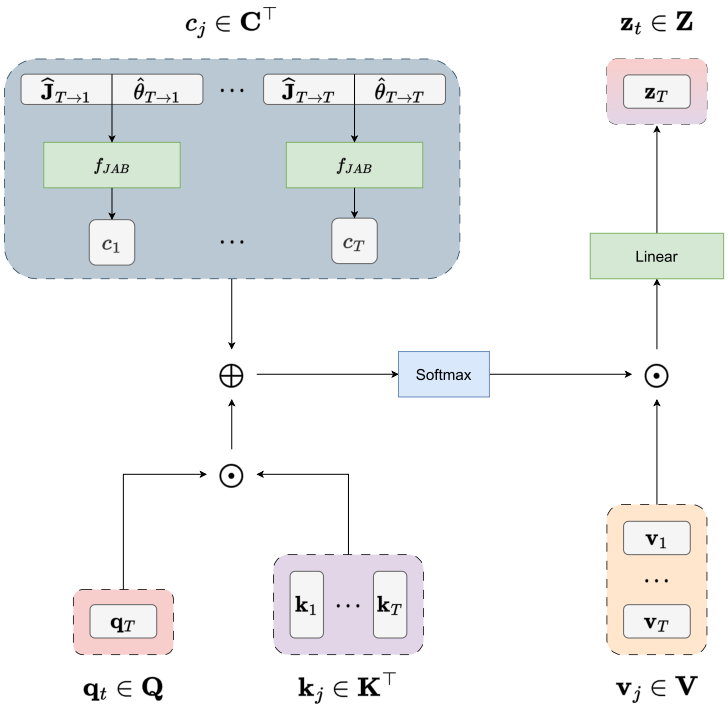


Figure 4.3: **Self Attention with the Joint-Aware Bias.** This figure depicts our proposed modification to the self-attention layer of f_{AR}. The query matrix $\boldsymbol{Q}$ is formed using only frame t, the most recent time step (in this case T), while the keys $\boldsymbol{K}$ and values $\boldsymbol{V}$ are formed from the $(t-w+1)$ most recent time steps. To compute the Joint-Aware Bias, we canonicalize the joints and poses of every step j with respect to the joints and poses of step t. The Joint-Aware Bias encoder, f_{JAB}, independently maps the canonicalized joints $(\widehat{\boldsymbol{J}}^n_{t\to j})$ and canonicalized pose $(\widehat{\boldsymbol{\theta}}^n_{t\to j})$ of each frame j to c_j, a scalar. The sequence of all c_j forms $\boldsymbol{C} \in \mathbb{R}^{(t-w+1)}$, which acts as a bias on the attention weights. The pre-softmax attention matrix is computed by taking the dot-product $(\odot)$ of $\boldsymbol{q}_t$ and every $\boldsymbol{k}_j$ to which the bias c_j is added and the result is passed through a softmax function to compute the attention weights $\boldsymbol{\alpha} \in \mathbb{R}^{1\times(t-w+1)}$. Finally the output matrix $\boldsymbol{Z}$ is computed in the same manner as traditional attention.

$$\begin{aligned} \boldsymbol{A} &= \frac{\boldsymbol{Q}\boldsymbol{K}^\top \oplus \boldsymbol{C}^\top}{\sqrt{d_k}} \quad \text{and} \\ \boldsymbol{\alpha} &= \text{softmax}\left(\boldsymbol{A}\right), \end{aligned} \tag{4.21}$$

where $\boldsymbol{A} \in \mathbb{R}^{1\times(t-w+1)}$ is the pre-softmax attention weights, $\boldsymbol{C} \in \mathbb{R}^{(t-w+1)}$ is our Joint-Aware Bias, and $\oplus$ represents elementwise addition. We illustrate this process in **Figure 4.3**.

(a) **Meshes from camera perspective** (b) **Canonicalized meshes**

Figure 4.4: **(a)** The meshes $\widehat{\Theta}_t^n$ and $\widehat{\Theta}_t^n$ projected to the image plane with $\widehat{\boldsymbol{H}}_t^n$ and $\widehat{\boldsymbol{H}}_j^n$ representing the corresponding transformation matrices that place them in the camera coordinate system. **(b)** A top-down view of the same meshes in the coordinate system defined by $(\widehat{\boldsymbol{H}}_t^n)^{-1}$, resulting in the pelvis of $\widehat{\Theta}_t^n$ being located at the origin and $\widehat{\Theta}_j^n$ being placed relative to it.

The Joint-Aware Bias, $\boldsymbol{C}$, is computed using the sequence of joints $\widehat{\boldsymbol{J}}^n$ and poses $\widehat{\boldsymbol{\theta}}^n$ passed to f_{AR}. Given j such that $w \leq j \leq t$, we canonicalize the joints to produce $\widehat{\boldsymbol{J}}_{t\to j}^n \in \mathbb{R}^{24\times 3}$. This represents the joints of person n at frame j relative to the pelvis of person n at frame t. Similarly, the canonicalized pose $\widehat{\boldsymbol{\theta}}_{t\to j}^n \in \mathbb{R}^{24\times 3\times 3}$ represents the relative rotation of the joints of person n in frame j with respect to the corresponding joints of person n in frame t. We use the Joint-Aware Bias encoder, f_{JAB}, to map the canonicalized joints and pose for frame j to the corresponding bias value with

$$f_{JAB} : \left\{ \widehat{\boldsymbol{J}}_{t\to j}^n,\ \widehat{\boldsymbol{\theta}}_{t\to j}^n \right\} \mapsto \boldsymbol{C}_j. \tag{4.22}$$

Joint Canonicalization

In **Figure 4.4** we show the effect of canonicalizing the meshes $\widehat{\Theta}_j^n$ and $\widehat{\Theta}_t^n$ with respect to the mesh from time t. While the Joint-Aware Bias uses canonicalized joints, we show

canonicalized meshes to make the result of the transformation more evident.

To compute the canonicalized joints $\widehat{\boldsymbol{J}}^n_{t\to j}$ we first determine the $SE(3)$ transformation matrix $\widehat{\boldsymbol{H}}^n_t \in \mathbb{R}^{4\times 4}$ using the global orientation of the pelvis joint in $\widehat{\boldsymbol{\theta}}^n_t$ and the global translation $\widehat{\boldsymbol{\gamma}}^n_t$. This matrix captures the Euclidean transformation that places the mesh $\widehat{\boldsymbol{\Theta}}^n_t$ in the camera coordinate system. The inverse of this matrix will undo the transformation and result in a mesh with the same pose $\widehat{\boldsymbol{\theta}}^n_t$ and shape $\widehat{\boldsymbol{\beta}}^n_t$ but with the pelvis located at the origin and aligned with the z-axis. We can then use the inverse transformation matrix $\left(\widehat{\boldsymbol{H}}^n_t\right)^{-1}$ to find the relative joints $\widehat{\boldsymbol{J}}^n_{t\to j}$ with

$$\left[\widehat{\boldsymbol{J}}^n_{t\to j}\ \mathbf{1}\right]^\top = \left(\widehat{\boldsymbol{H}}^n_t\right)^{-1}\left[\widehat{\boldsymbol{J}}^n_j\ \mathbf{1}\right]^\top, \tag{4.23}$$

where $\left[\widehat{\boldsymbol{J}}^n_j\ \mathbf{1}\right]^\top \in \mathbb{R}^{4\times 24}$ is the joints for frame j converted to homogeneous coordinates then transposed. To extract $\widehat{\boldsymbol{J}}^n_{t\to j} \in \mathbb{R}^{24\times 3}$ from the final product, we simply transpose the output matrix and drop the last column.

Pose Canonicalization

In addition to the canonicalized joints described above, we also compute the canonicalized pose $\widehat{\boldsymbol{\theta}}^n_{t\to j}$ to capture the relative rotation between each joint in the query pose $\widehat{\boldsymbol{\theta}}^n_t$ and key pose $\widehat{\boldsymbol{\theta}}^n_j$, as shown in **Figure 4.5**. Letting $\boldsymbol{R}^n_t$ and $\boldsymbol{R}^n_j$ represent the 3×3 rotation matrix of the same arbitrary joint for person n in frame t and frame j, we wish to find $\boldsymbol{R}^n_{t\to j}$ such that

$$\boldsymbol{R}^n_j = \boldsymbol{R}^n_{t\to j}\boldsymbol{R}^n_t, \tag{4.24}$$

where $\boldsymbol{R}^n_{t\to j}$ captures the rotation that needs to be applied to $\boldsymbol{R}^n_t$ to get $\boldsymbol{R}^n_j$. Considering that we have access to $\boldsymbol{R}^n_t$ and $\boldsymbol{R}^n_j$, we can simply rearrange **Equation 4.24** for $\boldsymbol{R}^n_{t\to j}$, giving us

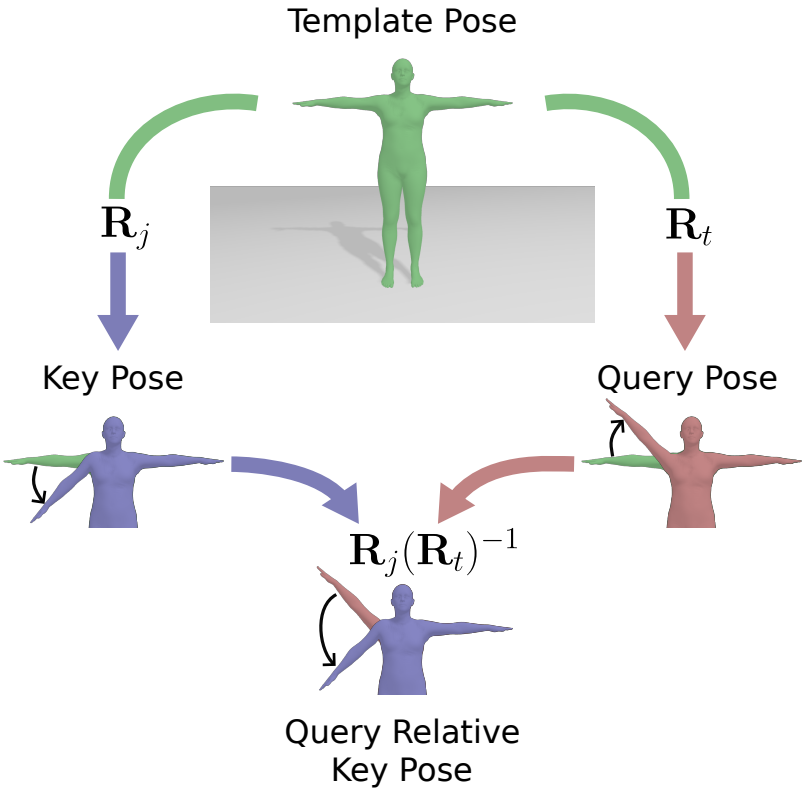


Figure 4.5: **Pose Canonicalization.** When dealing with pose, a rotation matrix parameterizes the rotation of some arbitrary joint with respect to a template pose. Here we show the template pose at the top, the key pose $\boldsymbol{R}_j$ on the left and the query pose $\boldsymbol{R}_t$ on the right representing two different rotations of the right shoulder joint. If we wish to canonicalize the key pose with respect to the query pose we need to compute $\boldsymbol{R}_{t\to j}$, which is simply the matrix product of the key pose and the inverse query pose. We can then use $\boldsymbol{R}_t$ as our new template pose meaning that we can compute $\boldsymbol{R}_j$ as $\boldsymbol{R}_j = \boldsymbol{R}_{t\to j}\boldsymbol{R}_t$.

$$\boldsymbol{R}^n_{t\to j} = \boldsymbol{R}^n_j \left(\boldsymbol{R}^n_t\right)^{-1}. \tag{4.25}$$

This relative rotation is computed for all 24 joints in $\widehat{\boldsymbol{\theta}}^n_j$, resulting in $\widehat{\boldsymbol{\theta}}^n_{t\to j} \in \mathbb{R}^{24\times 3\times 3}$.

Joint-Aware Bias Encoder

Once we have computed the relative joints $\widehat{\boldsymbol{J}}^n_{t\to j}$ and the relative pose $\widehat{\boldsymbol{\theta}}^n_{t\to j}$ for the entire input sequence, we flatten and concatenate the relative joints and poses for corresponding frames together to form $\boldsymbol{\Lambda} \in \mathbb{R}^{(t-w+1)\times 288}$ where

$$\mathbf{\Lambda} = \left[\begin{array}{c} flat(\widehat{\boldsymbol{J}}^n_{t\to w}) \\ \hdashline flat(\widehat{\boldsymbol{\theta}}^n_{t\to w}) \end{array}, \quad \cdots, \quad \begin{array}{c} flat(\widehat{\boldsymbol{J}}^n_{t\to t}) \\ \hdashline flat(\widehat{\boldsymbol{\theta}}^n_{t\to t}) \end{array} \right]^\top. \tag{4.26}$$

This is the input to our Joint-Aware Bias encoder, which is parameterized as an MLP. The Joint Aware Bias for every j such that $w \leq j \leq t$ is then computed at once with

$$\boldsymbol{C} = \sigma\left(\mathbf{\Lambda}\boldsymbol{W}_1\right)\boldsymbol{W}_2, \tag{4.27}$$

which consists of weights $\boldsymbol{W}_1 \in \mathbb{R}^{288\times576}$, $\boldsymbol{W}_2 \in \mathbb{R}^{576\times1}$, and σ represents the GELU activation function. Notably, the activation is applied only after the first linear layer, this is to ensure that the final output bias can positively or negatively influence the final attention weights. Once this bias is computed, we use $\boldsymbol{C} \in \mathbb{R}^{(t-w+1)}$ as described in **Equation 4.21**.

Autoregressive Prediction

Similar to PHD+T, predicting meshes more than a single frame into the future requires the model to run autoregressively. As described in **Equation 4.17** and **Equation 4.18**, f_{AR} will use data from step w to t to predict the mesh parameters for the next time step; t refers to the most recent time step and w is computed based on the size of the receptive field. For the first iteration f_{AR} uses information gleaned from frames 1 to $T = 25$ to estimate the future temporal representation $\widetilde{\mathbf{\Phi}}^n_{T+1}$ and latent joint representation $\widetilde{\mathbf{\Gamma}}^n_{T+1}$; we use these to generate the future SMPL parameters $\widetilde{\mathbf{\Theta}}^n_{T+1}$ from which we extract the joints $\widetilde{\boldsymbol{J}}^n_{T+1}$, poses $\widetilde{\boldsymbol{\theta}}^n_{T+1}$, and a new latent joint representation $\widetilde{\mathbf{\Gamma}}^n_{T+1}$. These newly extracted features are combined with the existing input for the next iteration of f_{AR} (where t now represents $T+1$) to predict features for $T+2$.

4.3.3 Training and Model Losses

Training this model follows the same two-step training procedure used by the original PHD model and our own PHD+T model. The first stage consists of training f_{movie} and f_{HMR} to produce accurate and consistent meshes based on observed frames. As this model also operates on each individual separately, we drop the person index n when describing the loss. The objective used to train both f_{movie} and f_{HMR} is

$$L_{movie} = \sum_{t=1}^{T+H} L_t + \lambda_{const} \sum_{t=1}^{T+(H-1)} \|\widehat{\boldsymbol{\beta}}_t - \widehat{\boldsymbol{\beta}}_{t+1}\|_2^2, \tag{4.28}$$

which is effectively the loss used to train f_{movie} in PHD+T (**Equation 4.9**) without the delta predictors term. After the performance of f_{movie} and f_{HMR} has converged, f_{AR} is trained using the exact same loss that was used to train the corresponding prediction head in PHD+T, namely **Equation 4.12**.

One key distinction in training f_{AR} that contrasts with traditional transformer based autoregressive models [65, 18] is that we eschew the use of teacher forcing [51]. Teacher forcing largely exists to allow for efficient training of large sequence models in a parallel fashion; it works by using ground-truth data as the input for all time steps of the sequence model while training. In contrast, when a trained model is used for inference, output predictions from the model are fed back as input to generate the next prediction autoregressively. This disconnect between training and inference is referred to as exposure bias [91] and it leads to errors accumulating in longer term predictions as the model has only seen relatively ideal data while training, as opposed to its own outputs that are likely much noisier [50]. Since f_{AR} is a fairly small transformer model with a single layer and one attention head, we decided to train it autoregressively in the hopes of mitigating the issue of error accumulation.

4.4 Multi-Person Joint-Aware Transformer

Our final model, the Multi-Person Joint-Aware Transformer (MPJAT), is an extension of the previous Joint-Aware Transformer (JAT); we build on it by designing it to model multiple people simultaneously and augmenting it with an additional loss that penalizes interpenetration of mesh vertices between different individuals. Inspired by the AgentFormer architecture [66], we modify the attention mechanism to contextualize the interactions of all individuals in a scene; we refer to this modified GPT layer as Agent-GPT. Since the goal of our work is to predict the future states of individuals in a video, leveraging the information available in the interactions between people is the intuitive next step to try and improve prediction performance.

4.4.1 Model Architecture

The fundamental difference in the architecture of the MPJAT and the JAT is that we swap out the attention mechanism in both f_{movie} and f_{AR} with the Agent-Aware attention mechanism proposed by Yuan *et al.* As shown in **Figure 4.6**, this allows us to feed the cropped image features $\boldsymbol{\phi}_t^n$ and the bounding box information $\boldsymbol{B}_t^n$ of all N individuals to f_{movie} at once. We denote this conglomerate input to f_{movie} as $\boldsymbol{X} \in \mathbb{R}^{Nt \times 2052}$ such that

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}^1 \\ \cdots \\ \boldsymbol{X}^N \end{bmatrix}, \tag{4.29}$$

with $\boldsymbol{X}^n$ defined as specified in **Equation 4.13**. Thus f_{movie} is a causal model that learns the function

$$f_{movie} : \boldsymbol{X} \mapsto \left[\widehat{\boldsymbol{\Phi}}_t^1, \ \cdots, \ \widehat{\boldsymbol{\Phi}}_t^N \right]^\top, \tag{4.30}$$

where $\left[\widehat{\boldsymbol{\Phi}}_t^1, \ \cdots, \ \widehat{\boldsymbol{\Phi}}_t^N \right]^\top \in \mathbb{R}^{N \times 512}$ and $\widehat{\boldsymbol{\Phi}}_t^n$ again denotes the temporal representation for

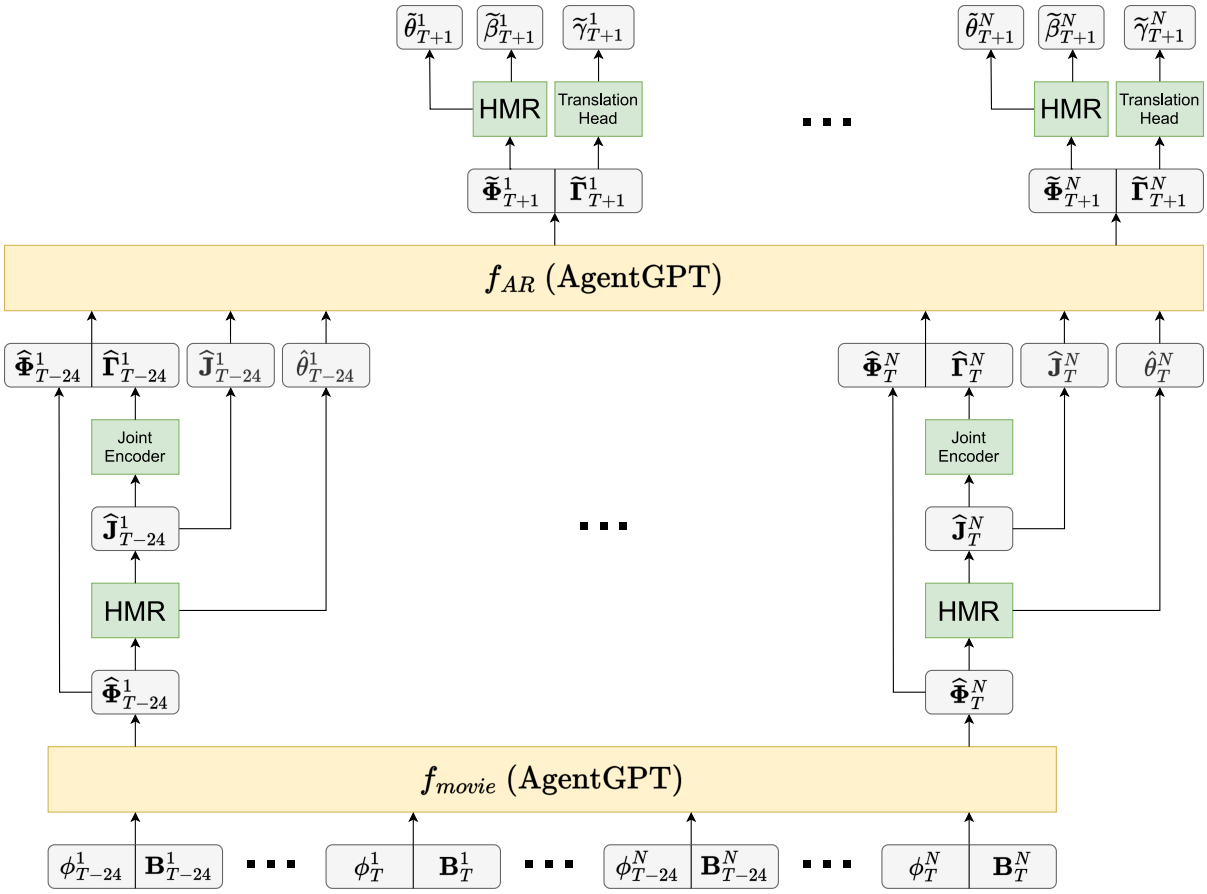


Figure 4.6: **The Multi-Person Joint-Aware Transformer Architecture.** MPJAT leverages Agent-Aware attention to simultaneously condition on and predict multiple people. In the same vein as our previous models there is a dedicated layer for building a temporal representation based on input frames, and a dedicated layer for prediction.

person n at frame t.

The inputs to f_{movie} are first positionally encoded and compressed in the same manner as they were in JAT; each individual n has the positional encoding applied to their corresponding input features independently as described by **Equation 4.15**. Since we use the positional encoding as a means to embed timestamps into the features, we apply them separately for each person to ensure that corresponding frames across different individuals are correlated. The encoded features for all N individuals are subsequently stacked in the same manner as $\boldsymbol{X}$, resulting in $\boldsymbol{E} \in \mathbb{R}^{Nt \times 512}$. These encoded features are passed to an AgentGPT layer to compute the temporal representations. Once the temporal representation $\widehat{\boldsymbol{\Phi}}_t^n$ for every person and frame is estimated by f_{movie}, we use

f_{HMR} to estimate the SMPL parameters $\widehat{\boldsymbol{\Theta}}_t^n$ and the joint encoder f_{JE} to compute the latent joint representation $\widehat{\boldsymbol{\Gamma}}_t^n$.

The AR head, f_{AR}, is also parameterized with an AgentGPT layer. In the same vein as the JAT, the primary input to f_{AR} for the MPJAT is the stacked sequences of concatenated temporal features $\widehat{\boldsymbol{\Phi}}^n$ and latent joint representations $\widehat{\boldsymbol{\Gamma}}^n$ of every individual. We denote this input as $\widehat{\boldsymbol{X}} \in \mathbb{R}^{N(t-w+1)\times 640}$ where

$$\widehat{\boldsymbol{X}} = \begin{bmatrix} \widehat{\boldsymbol{X}}^1 \\ \cdots \\ \widehat{\boldsymbol{X}}^N \end{bmatrix}, \tag{4.31}$$

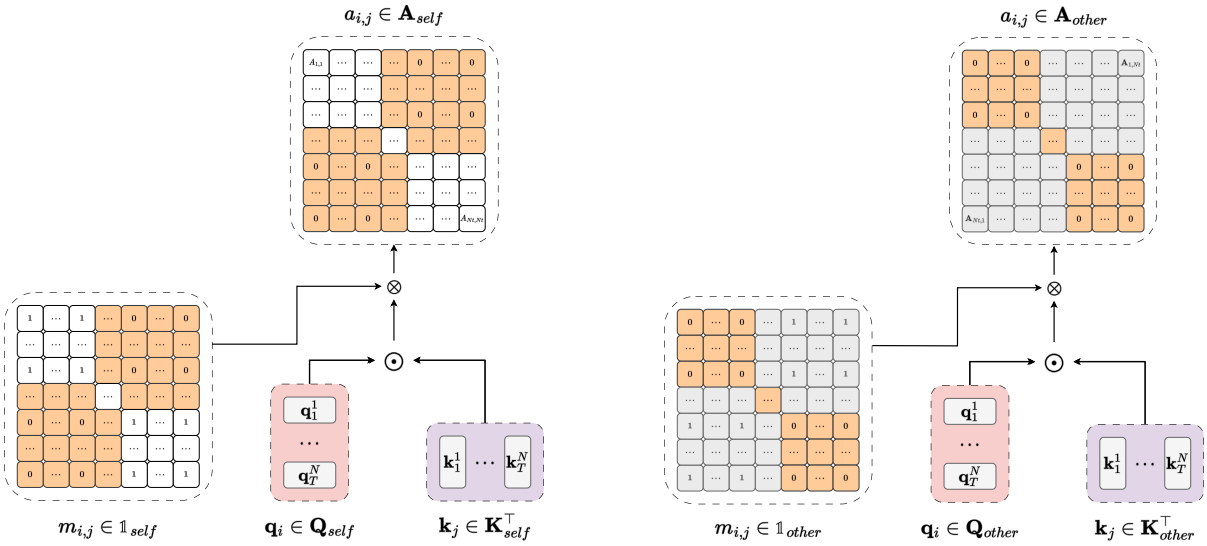
and $\widehat{\boldsymbol{X}}^n$ is structured as shown in **Equation 4.16**. Additionally, f_{AR} takes as input the 3D joint positions $\widehat{\boldsymbol{J}} \in \mathbb{R}^{N(t-w+1)\times 24\times 3}$ and rotations $\widehat{\boldsymbol{\theta}} \in \mathbb{R}^{N(t-w+1)\times 24\times 3\times 3}$ of all individuals combined in the same manner as $\widehat{\boldsymbol{X}}$. Given these inputs, f_{AR} models the future temporal representation $\widetilde{\boldsymbol{\Phi}}_{t+1}^n$ and latent joint representation $\widetilde{\boldsymbol{\Gamma}}_{t+1}^n$ for every person as

$$f_{AR} : \left\{ \widehat{\boldsymbol{X}},\ \ \widehat{\boldsymbol{J}},\ \ \widehat{\boldsymbol{\theta}} \right\} \mapsto \left[\begin{array}{c:c:c} \widetilde{\boldsymbol{\Phi}}_{t+1}^1 & \cdots & \widetilde{\boldsymbol{\Phi}}_{t+1}^N \\ \widetilde{\boldsymbol{\Gamma}}_{t+1}^1 & & \widetilde{\boldsymbol{\Gamma}}_{t+1}^N \end{array} \right]^\top. \tag{4.32}$$

Before passing $\widehat{\boldsymbol{X}}$ to the AgentGPT layer of f_{AR}, we encode each $\widehat{\boldsymbol{X}}^n$ individually; for each person n we use **Equation 4.19** to compute the encoded features $\widehat{\boldsymbol{E}}^n$ and then stack them to produce

$$\widehat{\boldsymbol{E}} = \begin{bmatrix} \widehat{\boldsymbol{E}}^1 \\ \cdots \\ \widehat{\boldsymbol{E}}^N \end{bmatrix}, \tag{4.33}$$

where $\widehat{\boldsymbol{E}} \in \mathbb{R}^{N(t-w+1)\times 640}$. The additional inputs $\widehat{\boldsymbol{J}}$ and $\widehat{\boldsymbol{\theta}}$ are used to compute the canonicalized joints $\widehat{\boldsymbol{J}}_{t\to j}^{n\to o}$ and canonicalized poses $\widehat{\boldsymbol{\theta}}_{t\to j}^{n\to o}$. These canonicalized inputs are



(a) **Agent-Aware Attention - Self** (b) **Agent-Aware Attention - Other**

Figure 4.7: Computation of the attention weights in Agent-Aware attention is split into two different sets of query and key matrices where $\odot$ refers to matrix multiplication, $\otimes$ is elementwise multiplication, and orange elements denote masked out values. **(a)** $\boldsymbol{A}_{self}$ is computed using only queries and keys that refer to the same individual. **(b)** $\boldsymbol{A}_{other}$ consists only of query-key pairs where the queries and keys do not stem from the same individual.

in turn used to compute our Joint-Aware Bias, which we generalize to the multi-person case. In the next section we describe Agent-Aware attention, and in the subsequent section we expand on how it would be combined with our Joint-Aware Bias.

4.4.2 Agent-Aware Attention

Our primary interest in exploring a multi-person transformer is to model the interactions between individuals and use this to inform our predictions. For example, when using f_{movie} to compute our temporal representation $\widehat{\boldsymbol{\Phi}}_t^n$ we want to consider all input features $\boldsymbol{X}^n$ specific to individual n and simultaneously we want to account for the salient information available in the input features of any other individual o embedded in $\boldsymbol{X}^o$. The Agent-Aware attention mechanism proposed by Yuan *et al.* [66] does exactly that by modelling both spatial and temporal information in the input features concurrently. It does so by learning two different sets of query and key matrices, as seen in **Figure 4.7**.

The first set of query and key matrices, denoted as $\boldsymbol{Q}_{self}$ and $\boldsymbol{K}_{self}$, are computed using only the frames from a person of interest n. The second set of query and key matrices, denoted as $\boldsymbol{Q}_{other}$ and $\boldsymbol{K}_{other}$, are computed using only the frames from every other individual o such that $o \neq n$. These matrices are computed over the entire set of inputs in the same manner as described in **Equation 3.16**, where $\boldsymbol{Q}_{self} \in \mathbb{R}^{Nt \times d_k}$, $\boldsymbol{K}_{self} \in \mathbb{R}^{Nt \times d_k}$ and the corresponding $\boldsymbol{Q}_{other}$, $\boldsymbol{K}_{other}$ are of the same shape. The purpose of the extra set of queries and keys is to better disambiguate the entries in $\boldsymbol{\alpha} \in \mathbb{R}^{Nt \times Nt}$ that correspond to an individual attending to their own past v.s. the entries where they're attending to another person's past. The final attention weights $\boldsymbol{\alpha}$ can be calculated with

$$\begin{aligned} \boldsymbol{A}_{self} &= \mathbb{1}_{self} \otimes \frac{\boldsymbol{Q}_{self}\boldsymbol{K}_{self}^{\top}}{\sqrt{d_k}}, \\ \boldsymbol{A}_{other} &= \mathbb{1}_{other} \otimes \frac{\boldsymbol{Q}_{other}\boldsymbol{K}_{other}^{\top}}{\sqrt{d_k}}, \\ \boldsymbol{\alpha} &= \text{softmax}\left(\boldsymbol{A}_{self} \oplus \boldsymbol{A}_{other} \oplus \boldsymbol{M}\right), \end{aligned} \tag{4.34}$$

where $\otimes$ refers to elementwise multiplication, $\oplus$ is elementwise addition, and $\mathbb{1}_{self}$, $\mathbb{1}_{other}$ are indicator functions that denote which elements of the attention weights correspond to the 'self' matrices and which ones correspond to the 'other' matrices. Letting i, j represent the row and column index of $\mathbb{1}_{self}$ respectively with $i, j \in \{1 \cdots Nt\}$, we can compute $\mathbb{1}_{self}$ as

$$(\mathbb{1}_{self})_{i,j} = \begin{cases} 1 & \text{if } \lfloor \frac{i-1}{t} \rfloor = \lfloor \frac{j-1}{t} \rfloor, \\ 0 & \text{otherwise.} \end{cases} \tag{4.35}$$

The indicator function for the 'other' attention weights, $\mathbb{1}_{other}$, is simply the opposite of the above where

$$(\mathbb{1}_{other})_{i,j} = 1 - (\mathbb{1}_{self})_{i,j}. \tag{4.36}$$

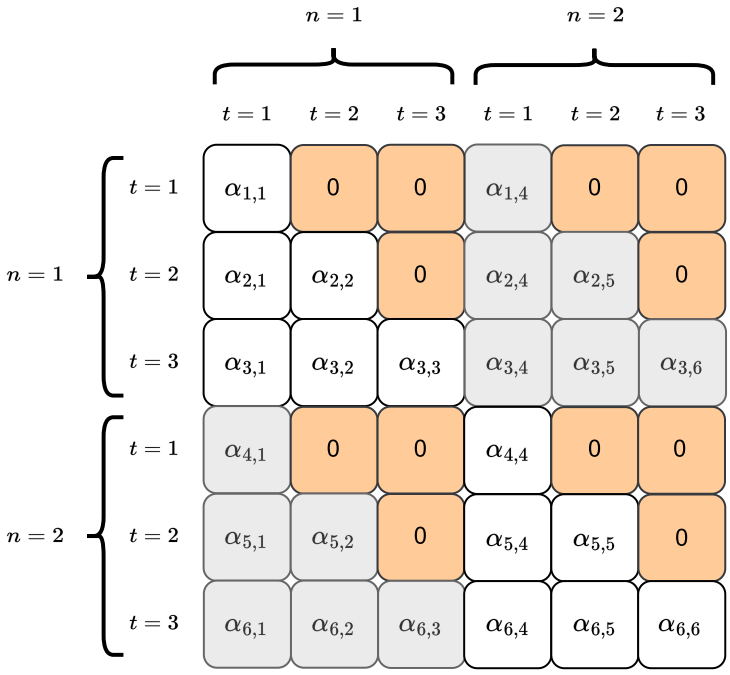


Figure 4.8: **Agent-Aware Attention Weights** The final attention weights $\boldsymbol{\alpha}$ computed using Agent-Aware attention after causal masking where $N = 2$ and $t = 3$. The top left and bottom right $t \times t$ quadrants in white encode the relationships between an individual's own features, for example the top left captures the temporal dependencies of person one's own features across time. Similarly, the top right and bottom left quadrants in gray encode how the individuals in the scene influence each other.

Finally, a causal mask $\boldsymbol{M} \in \mathbb{R}^{Nt \times Nt}$ is applied to the pre-softmax attention weights to ensure future information is blocked out from both the 'self' and 'other' components; it is constructed by tiling the standard $t \times t$ mask N times in both directions. The structure of the final attention weights $\boldsymbol{\alpha}$ after application of softmax can be seen in **Figure 4.8**. Computing the final output then proceeds the same as always as there is only a single value matrix $\boldsymbol{V}$ and the rest of the process is unchanged.

4.4.3 Agent-Aware Attention and our Joint-Aware Bias

While Agent-Aware attention as we described above is sufficient enough to communicate how f_{movie} leverages it to generate our temporal representation $\widehat{\boldsymbol{\Phi}}_t^n$, we need to make some minor modifications so we can use it alongside our Joint-Aware Bias to predict future meshes with f_{AR}. We build on the idea of having separate query and key matrices for

quantifying the dependency between the same and distinct individuals by additionally training separate Joint-Aware Bias encoders for computing the bias between the relative poses of the same individual and different individuals. First, we reformulate **Equation 4.23** as

$$\left[\widehat{\boldsymbol{J}}_{t\to j}^{n\to o}\ \mathbf{1}\right]^{\top} = \left(\widehat{\boldsymbol{H}}_{t}^{n}\right)^{-1}\left[\widehat{\boldsymbol{J}}_{j}^{o}\ \mathbf{1}\right]^{\top}, \tag{4.37}$$

where $\widehat{\boldsymbol{J}}_{t\to j}^{n\to o}$ represents the joints of person o at frame j in the coordinate space of person n at frame t. Similarly, **Equation 4.24** becomes

$$\boldsymbol{R}_{t\to j}^{n\to o} = \boldsymbol{R}_{j}^{o}\left(\boldsymbol{R}_{t}^{n}\right)^{-1}, \tag{4.38}$$

with $\boldsymbol{R}_{t\to j}^{n\to o}$ corresponding to the relative rotation of some arbitrary joint of person o at frame j with respect to the same joint of person n at frame t. These are used to simultaneously compute two Joint-Aware Biases, $\boldsymbol{C}_{self} \in \mathbb{R}^{N\times N(t-w+1)}$ and $\boldsymbol{C}_{other} \in \mathbb{R}^{N\times N(t-w+1)}$. We compute $\boldsymbol{C}_{self}$ using the Joint-Aware Bias encoder f_{JABS}, it maps

$$f_{JABS} : \left\{\widehat{\boldsymbol{J}}_{t\to j}^{n\to o},\ \widehat{\boldsymbol{\theta}}_{t\to j}^{n\to o}\right\} \mapsto (\boldsymbol{C}_{self})_{n,g}, \tag{4.39}$$

with n corresponding to the row and $g = \big((o-1)\times(t-w+1)\big)+j$ representing the column index. Similarly, we compute $\boldsymbol{C}_{other}$ in the same manner using another Joint-Aware Bias encoder we denote f_{JABO}; both encoders follow the one described in **Equation 4.27**, but have their own dedicated set of weights. Once we compute the specified biases, we can calculate the final attention weights with

$$\begin{aligned}
\boldsymbol{A}_{self} &= \mathbb{1}_{self} \otimes \frac{\boldsymbol{Q}_{self}\boldsymbol{K}_{self}^{\top} \oplus \boldsymbol{C}_{self}}{\sqrt{d_k}}, \\
\boldsymbol{A}_{other} &= \mathbb{1}_{other} \otimes \frac{\boldsymbol{Q}_{other}\boldsymbol{K}_{other}^{\top} \oplus \boldsymbol{C}_{other}}{\sqrt{d_k}}, \\
\boldsymbol{\alpha} &= \text{softmax}\left(\boldsymbol{A}_{self} \oplus \boldsymbol{A}_{other}\right),
\end{aligned} \tag{4.40}$$

where the indicator functions $\mathbb{1}_{self}$, $\mathbb{1}_{other}$ will again gate the irrelevant portions of $\boldsymbol{C}_{self}$ and $\boldsymbol{C}_{other}$. The attention weights are $\boldsymbol{\alpha} \in \mathbb{R}^{N \times N(t-w+1)}$ where we have N rows as our queries consist only of latest frame t for each person n (see **Section 4.3.2**).

4.4.4 Training and Model Losses

Training the MPJAT model is virtually identical to training the JAT model; the model we just put forth is also trained autoregressively as described in **Section 4.3.3**. The primary difference is that we now include the interpenetration loss described in [19]. This loss serves to ensure that two individuals cannot occupy the same physical space simultaneously. It is computed by generating a signed distance field for each person n in frame t where

$$\rho_t^n(x, y, z) = -\min(\text{SDF}(x, y, z), 0), \tag{4.41}$$

will return 0 if some 3D point (x, y, z) exists outside the mesh, or it will return a positive value relative to how far into the mesh of person n the point lies. We denote the total interpenetration loss for frame t as

$$P_t = \sum_{n=1}^{N} \sum_{o=1, o \neq n}^{N} \sum_{v \in \boldsymbol{V}_t^o} \rho_t^n(v) \tag{4.42}$$

where $v \in \boldsymbol{V}_t^o$ corresponds to the vertices sampled from the mesh of individual o at frame t. We sum this loss over all possible n and o, including the cases where the individuals

have just switched indexes. With this addition, the loss we use to train f_{movie} is now

$$L_{movie} = \sum_{t=1}^{T+H} [L_t + P_t] + \lambda_{const} \sum_{t=1}^{T+(H-1)} \|\widehat{\boldsymbol{\beta}}_t - \widehat{\boldsymbol{\beta}}_{t+1}\|_2^2, \tag{4.43}$$

and similarly, the loss for f_{AR} becomes

$$\begin{aligned} L_{AR} = \sum_{t=T+1}^{T+H} \left[L_t + P_t + \lambda_{temp} \|\widetilde{\boldsymbol{\Phi}}_t - \widehat{\boldsymbol{\Phi}}_t\|_2^2 + \lambda_{transl} \|\widetilde{\boldsymbol{\gamma}}_t - \widehat{\boldsymbol{\gamma}}_t\|_2^2 \right] \\ + \lambda_{const} \sum_{t=T+1}^{T+(H-1)} \|\widetilde{\boldsymbol{\beta}}_t - \widetilde{\boldsymbol{\beta}}_{t+1}\|_2^2. \end{aligned} \tag{4.44}$$

Chapter 5

Experiments

In this section we discuss our experiments to evaluate the efficacy of our proposed ideas. We start by discussing the datasets we chose, describe our evaluation methodology, and finally provide our results.

5.1 Datasets

We train and evaluate the performance of our proposed models on Human3.6M [92] and 3DPW [93], two standard datasets in the human pose estimation space. In **Table 5.1** we describe the sample counts for the train, validation, and test splits of both datasets after pre-processing. Since the test split of 3DPW is almost twice the size of its train split and it has almost 3 times the number of multi-person samples, we switch them. That is, only for 3DPW, we train on the test split and test on the train split.

Human3.6M

Human3.6M [92] is a large scale, single-person, video dataset consisting of RGB video. This dataset has seven different subjects, each of them independently recorded enacting 30 different scenes in a motion capture system with four different camera views. It provides 2D & 3D poses, bounding boxes, and depth maps. However, Human3.6M does

Datasets				
Dataset	Split	Num Single Person	Num Multi-Person	Num Total
H3.6M	train	134,722	0	134,722
	val	47,010	0	47,010
	test	63,652	0	63,652
3DPW	train	12,042	4,635	16,677
	val	5,346	1,970	7,316
	test	12,842	11,254	24,096

Table 5.1: **Datasets.** The total number of samples for the train,validation, and test splits of both datasets we are using in this work. Due to the limited size of the train split of 3DPW we use train on the test split and test on train split.

not include SMPL parameters, so we first generate pseudo-groundtruth SMPL parameters with a version of SMPLify that is modified to fit a mesh to 3D keypoints [19]. Following previous work [21, 7, 19] we use protocol 2 to define our train and test splits; we use only the main camera view and subjects S1, S6, S7, S8 form our train split, S5 is our validation split, and S9, S11 are our test split.

3DPW

3D Poses in the Wild (3DPW) [93] is a dataset consisting of various outdoor and indoor scenes containing up to two people captured with a dynamic camera and inertial measurement units strapped to individuals' limbs; after capturing the initial data the mesh parameters and extrinsic camera matrix were computed through the use of a continuous optimization framework. Due to the dynamic camera, the dataset provides the SMPL parameters and the 3D keypoints in the world coordinate frame. Our models however perform estimation and prediction in the camera coordinate frame. This, in addition to the fact that the dataset lacks groundtruth bounding boxes and parameterizes 2D keypoints using a different set of joints than the 3D keypoints, requires us to preprocess the dataset before use. This preprocessing entails transforming the 3D mesh parameters and 3D keypoints from the world coordinate frame to the camera coordinate frame, projecting

the 3D keypoints to the image plane to compute corresponding 2D keypoints, and then using the 2D keypoints to generate bounding boxes. We primarily use 3DPW to quantify how well MPJAT performs on multiple people when compared against our single-person methods. As evidenced by **Table 5.1** the training split not only has a limited amount of multi-person samples, but it has less samples over all than the test split. For this reason we switch the two splits.

5.2 Implementation Details

For all of our models, we follow the work in PHD and use a ResNet-50 backbone [94] initialized with the learned weights from HMR. The backbone for every model is frozen during all stages of training. We first train PHD+T and JAT on the Human3.6M dataset for conditioning. After convergence we freeze the weights of the temporal conditioning model f_{movie} and the HMR head f_{HMR} before proceeding to train on the prediction task for Human3.6M.

When we train on 3DPW we follow the same process where we first train on conditioning, then prediction. When training for the conditioning task we bootstrap the models with the f_{movie} and f_{HMR} weights from the corresponding Human3.6M model; in the case of MPJAT we use JAT's weights and duplicate the necessary sub-models for Agent-Aware attention. After all three models have been finetuned on 3DPW for conditioning, we freeze the relevant parts and load in the f_{AR} weights from the corresponding Human3.6M models to train for prediction.

To increase data variety, we leverage data augmentation while training. Our data augmentation pipeline follows HMMR [7] and PHD [12]; it consists of jittering the input bounding boxes to simulate noisy tracking models, and randomly flipping the entire input video. All models are trained with a batch size of 32 on a single NVIDIA V100 GPU. The largest bottleneck in our data-loading pipeline is the cropping performed for

each bounding box. Similarly, a large part of GPU memory is dedicated to storing the backbone weights and corresponding backbone features. Since all of our models share a common frozen backbone, we precompute the backbone features for all crops, eliminating the need to crop and compute features during training. All together, we jitter the bounding boxes, perform the random flip, crop out the bounding boxes, and then pass the crop to the backbone and store the backbone outputs on disk. To maintain data variety, we cache eight epochs worth of data and randomly sample from it during training.

Finally, we present the hyperparameters we used for training in **Appendix C**. We use the Hyperband [95] algorithm to find the best performing hyperparameters for PHD+T and JAT on the conditioning and prediction tasks of Human3.6M. We then use the same hyperparameters when we train on 3DPW, where we reuse the tuned hyperparameters for JAT on the MPJAT model.

5.3 Evaluation

In this section we first provide an overview of our evaluation process, describe the metrics we are using, and then share and discuss our results. We evaluate the models for both conditioning and prediction, where we refer to conditioning as the process through which the model reconstructs the observed input frames and refer to prediction as the stage where it estimates the 3D meshes of future unseen frames.

For the conditioning stage, since the PHD+T model cannot reconstruct the first 12 frames (due to its receptive field of $r = 13$), while the transformer based models can, we evaluate conditioning strictly on the frames that are reconstructed by all methods. For the prediction stage, we follow the original PHD [12] work in our evaluation process. While training, every model is fed $T = 25$ input frames and predicts the next 25 future frames. When we test our models, we again feed them $T = 25$ input frames, but now

we make them predict 30 frames to evaluate how well it can predict frames over the sequence length seen during training. As an additional baseline to evaluate prediction performance, we introduce the ZeroMotion baseline. The ZeroMotion baseline simply repeats the estimated meshes of the last observed frame for all future frames. These meshes are estimated using f_{movie} and f_{HMR} from the $PHD + T$ model.

5.3.1 Metrics

We evaluate on three common metrics in the 3D mesh estimation space, described below.

Mean Per Joint Position Error (MPJPE): This metric measures the average 3D distance in millimeters (mm) between corresponding joints of the predicted and groundtruth poses after aligning the root joint. The closer this measure is to zero, the more similar the two poses are.

Procrustes Aligned Mean Per Joint Position Error (PA-MPJPE): Like above, this metric measures the 3D distance between corresponding joints in millimeters. The difference is that after aligning the root joint, the poses are also aligned using Procrustes Analysis [96] which scales and rigidly rotates the prediction to best match the groundtruth. This metric strictly focuses on pose similarity while the prior metric also accounts for global rotation and size of the mesh. Similar to MPJPE, the closer this measure is to zero, the more similar the two poses are.

Percent Correct Keypoints (PCK): The final metric quantifies how similar the predicted and groundtruth poses are after being projected to the 2D image plane. After computing the pixel distance between the corresponding joints, we find the percent of predicted joints that are within $\alpha \times \max(h, w)$ pixels of the groundtruth, where $\alpha = 0.05$ and h, w represent the height and width of the bounding box respectively. While previous measures align at the pelvis, this one does not, meaning

Human3.6M - Conditioning			
Arch.	PA-MPJPE ↓	MPJPE ↓	PCK ↑
PHD+T	61.9	88.8	81.0
JAT	61.7	86.6	81.8

Table 5.2: **Conditioning Performance on Human3.6M.** For all metrics the JAT performs slightly better than PHD+T. At this point, the primary difference is in the use of the transformer model for f_{movie}, meaning it potentially offers a better approach to temporal aggregation.

that the predicted location of the body in the scene will affect this score. The closer this measure is to 100, the more similar the prediction is to the groundtruth.

5.3.2 Performance on Human3.6M

We start by discussing results on the Human3.6M dataset. In **Appendix D.1** we show the predictions of the PHD+T and JAT models on samples from the Human3.6M test set. Since Human3.6M consists of samples recorded in a motion capture studio with a static camera, it represents a fairly elementary challenge. The background is easy to disentangle from the individual being captured and the lack of a moving camera means that all motion can be directly attributed to movement of the person around the scene.

Conditioning

Table 5.2 showcases how the PHD+T and JAT models perform on Human3.6M for conditioning. It can be seen that our proposed JAT model performs slightly better than PHD+T across the board. As this is only the conditioning stage, the Joint-Aware Bias is not yet relevant; these results only tell us that using a standard GPT-3 style transformer in lieu of a TCN can beget positive results on the conditioning task.

We performed an ablation on the 6D rotation representation for PHD+T to quantify the difference between it and axis-angle rotations, which we show in **Table 5.3**. Both transformer models we propose make use of the 6D rotation representation, described in

Human3.6M - Conditioning				
Arch.	6D	PA-MPJPE ↓	MPJPE ↓	PCK ↑
PHD+T	-	66.7	106.1	74.7
PHD+T	✓	61.9	88.8	81.0

Table 5.3: **Ablation on 6D Rotation Representation.** We run an ablative experiment on the conditioning task of Human3.6M to measure the difference in performance between axis-angle and the 6D rotation representation for the PHD+T architecture. As described in **Section 3.2.3** this disparity in performance is primarily due to there being a many-to-one mapping between axis-angle rotations and equivalent rotation matrices, resulting in axis-angle being a noisy parameterization to regress.

	Human3.6M - Prediction														
	PA-MPJPE ↓					MPJPE ↓					PCK ↑				
Arch.	1	5	10	20	30	1	5	10	20	30	1	5	10	20	30
ZeroMot	63.7	75.2	88.4	100.9	103.0	90.8	105.0	124.8	153.1	171.5	79.3	65.2	52.5	40.5	34.7
PHD+T	63.0	71.5	84.1	99.2	109.6	89.8	100.0	118.6	149.9	173.4	79.6	69.7	56.8	38.7	29.7
JAT	61.5	69.9	78.1	89.0	97.8	87.0	100.2	115.0	137.8	157.0	76.6	66.7	56.7	42.8	30.8

Table 5.4: **Prediction Performance on Human3.6M.** Performance metrics for both models and the ZeroMotion baseline averaged over specified future frames. JAT has the lowest PA-MPJPE and MPJPE, telling us that it can most accurately predict pose. Conversely, PHD+T has the highest PCK measure in the short term with JAT performing best on PCK at 20 frames into the future and ZeroMotion performing best on PCK at 30 frames into the future.

Appendix B. The original PHD model however uses axis-angle as its rotation representation. It can be seen that the use of the 6D rotation representation has an outsized effect on model performance. For this reason, we use the 6D rotation representation in our PHD+T model to ensure a valid comparison between the transformer models and PHD+T. This ensures that any difference in performance is directly due to how temporal information is modelled and not because of an ineffective rotation parameterization.

Prediction

Table 5.4 displays the performance of our proposed models at the prediction task on Human3.6M. We also plot these results in **Figure 5.1** to make the results easier to parse. From these results, the JAT model plainly performs best in regards to PA-MPJPE and

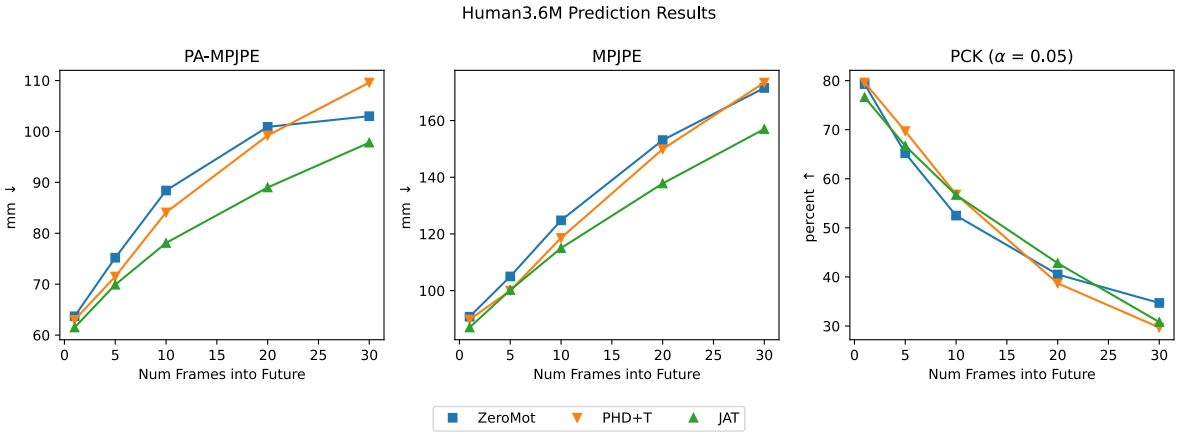


Figure 5.1: **Human3.6M Prediction Performance Plot.** Plots of prediction performance on Human3.6M averaged over specific future frames for each metric. For PA-MPJPE and MPJPE lower values indicate better performance, for PCK higher values indicate better performance. JAT outperforms PHD+T and the ZeroMotion baseline on PA-MPJPE and MPJPE across almost all future frames but PHD+T performs best early on for PCK while ZeroMot dominates PCK performance at 30 frames into the future.

MPJPE; for PCK however, PHD+T is more accurate in the short-term while JAT is more accurate further into the future with the ZeroMotion baseline performing the best at 30 frames into the future. These results indicate that JAT predicts the pose dynamics and long term location more accurately than PHD+T.

As an ablation to validate the effectiveness of the Joint-Aware Bias, we train a version of JAT without the bias and show the results in **Table 5.5**. It quickly becomes evident that using our proposed Joint-Aware Bias is beneficial as it improves performance on all metrics with respect to a transformer model without the Joint-Aware Bias. This can potentially be due to the ego-centric representation used by the Joint-Aware Bias helping to better model trajectory, since the largest improvement is in PCK.

5.3.3 Performance on 3DPW

We now discuss the results of our models on the 3DPW dataset, which is a significantly more difficult dataset compared to Human3.6M. The 3DPW dataset consists of in-the-wild samples with a dynamic camera; all of our models predict the location of each

		Human3.6M - Prediction														
		PA-MPJPE ↓					MPJPE ↓					PCK ↑				
Arch.	J-Bias	1	5	10	20	30	1	5	10	20	30	1	5	10	20	30
JAT	-	61.8	70.2	78.9	90.2	98.5	87.1	100.4	116.2	140.1	158.5	75.4	63.5	53.2	40.8	27.8
JAT	✓	61.5	69.9	78.1	89.0	97.8	87.0	100.2	115.0	137.8	157.0	76.6	66.7	56.7	42.8	30.8

Table 5.5: **Ablation on Joint-Aware Bias.** Our proposed Joint-Aware Bias improves performance of the transformer model across all metrics. The largest improvement can be seen in the PCK measure, indicating that our Joint-Aware bias helps to better predict the trajectory of the individual in question.

individual in the camera coordinate frame. This means that for scenes with significant camera motion, the models must jointly predict the camera motion and the individuals' trajectories. A better approach would be to estimate the camera motion independently, and then use it to predict each individuals' trajectories in world coordinates, but we leave this as future work.

In **Appendix D.2** we show our model predictions on 3DPW samples. With 3DPW we are primarily interested in evaluating how our models perform at predicting human motion in scenes with more than one person. As such, the results we present here are computed using only samples from our test split (3DPW's train split) that contain two individuals, since 3DPW does not have any scenes with more than two people. For the sake of completeness however, we discuss results over all test samples in **Section 5.4**.

Conditioning

In **Table 5.6** we present how well our models perform at reconstructing observed meshes on our 3DPW test set. We can see that all three models perform roughly the same with PHD+T having the best performance on the PA-MPJPE and MPJPE, which tells us that the PHD+T model is slightly more accurate at predicting the pose of observed people. On the other hand, MPJAT has a slightly higher PCK score meaning that it might be doing a better job of modelling the trajectories of observed individuals.

As an ablation experiment, we evaluate the effectiveness of the interpenetration loss

3DPW - Conditioning (Multi-Person Samples)			
Arch.	PA-MPJPE ↓	MPJPE ↓	PCK ↑
PHD+T	81.1	121.0	66.6
JAT	85.5	125.3	67.0
MPJAT	84.0	123.9	67.5

Table 5.6: **Conditioning Performance on 3DPW (Multi-Person Samples).** When it comes to PA-MPJPE and MPJPE, all models perform relatively the same with there being about a 4 mm difference between the best performing model (PHD+T) and the worst performing model (JAT). When considering PCK, MPJAT similarly performs roughly 1% better than PHD+T with JAT being in the middle.

3DPW - Conditioning (Multi-Person Samples)				
Arch.	P_t	PA-MPJPE ↓	MPJPE ↓	PCK ↑
JAT	-	85.5	125.3	67.0
JAT	✓	82.5	124.8	64.3
MPJAT	-	86.4	128.4	65.8
MPJAT	✓	84.0	123.9	67.5

Table 5.7: **Ablation on P_t for Conditioning (Multi-Person Samples).** We train our proposed single-person JAT model with the interpenetration loss P_t and the multi-person MPJAT model without as an ablative experiment to quantify the usefulness of P_t. Using the loss improves PA-MPJPE and MPJPE for JAT but leads to degenerate PCK performance. MPJAT however improves performance for all metrics with P_t.

P_t (**Equation 4.42**). We specifically propose P_t for the multi-person model, MPJAT. We evaluate both, disabling P_t for MPJAT and training the single-person transformer model, JAT, with it. We attempted to train PHD+T with the interpenetration loss as well, but the time-per-iteration was too high, ultimately making it intractable. We can see in **Table 5.7** that compared to training JAT without P_t, training JAT with it improves performance on PA-MPJPE and MPJPE, but compromises performance on PCK. MPJAT with P_t however shows improved performance for all metrics when compared against the same model without P_t. This can potentially be due to the Agent-Aware attention mechanism allowing MPJAT to better model occupancy of multiple people simultaneously. During training, the signal from the interpenetration loss encourages MPJAT to arrange the people in a manner that best fits the observation without leading to inter-

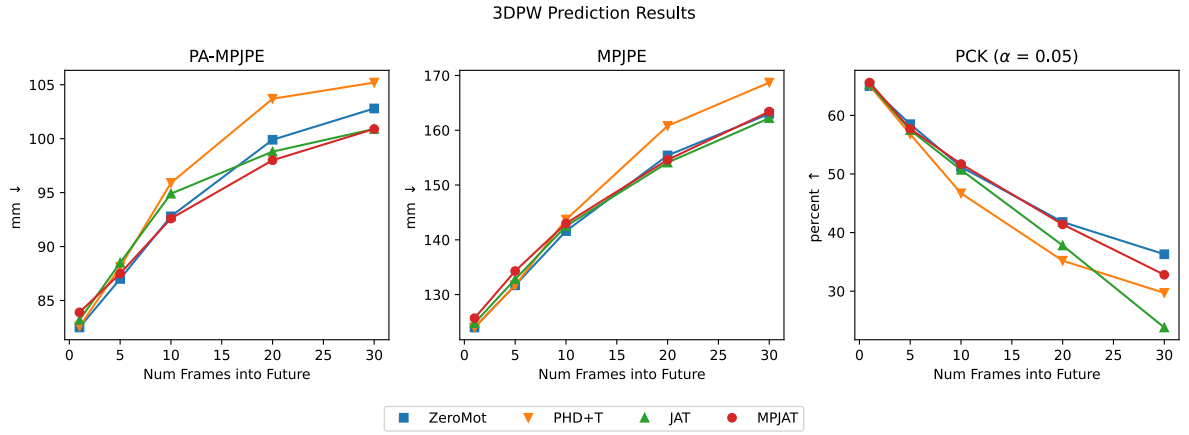


Figure 5.2: **3DPW Multi-Person Prediction Performance Plot.** Plots of prediction performance on 3DPW averaged over specific future frames for each metric. Generally speaking, MPJAT performs best on PA-MPJPE with JAT and ZeroMotion closely following while the three models have comparable performance on MPJPE. In regards to PCK, MPJAT and ZeroMotion perform roughly the same up until 30 frames into the future where ZeroMotion is best.

sections of the meshes, JAT on the other hand has no way to corroborate the location of other meshes in the scene.

Prediction

Figure 5.2 plots the prediction performance of our models with **Table 5.8** being the corresponding table. From these we can see that MPJAT and the ZeroMotion baseline have the highest PCK scores. The two of them are fairly comparable for PCK performance until 30 frames into the future, where the ZeroMotion baseline performs best. This indicates that when it comes to the multi-person subset of the 3DPW dataset, MPJAT and ZeroMotion can model the trajectories of each person significantly better than PHD+T and the single-person transformer model, JAT. When looking at PA-MPJPE and MPJPE we can see that the ZeroMotion baseline performs better on earlier frames while on later frames MPJAT scores higher on PA-MPJPE and JAT scores higher on PA-MPJPE. This implies that both MPJAT and JAT can model long-term pose dynamics better in the multi-person case.

	3DPW - Prediction (Multi-Person Samples)														
	PA-MPJPE ↓					MPJPE ↓					PCK ↑				
Arch.	1	5	10	20	30	1	5	10	20	30	1	5	10	20	30
ZeroMot	82.5	87.0	92.8	99.9	102.8	124.0	131.7	141.6	155.4	163.0	65.0	58.5	51.2	41.8	36.3
PHD+T	82.6	88.1	95.9	103.7	105.2	124.0	131.7	143.7	160.8	168.7	65.1	56.8	46.7	35.2	29.7
JAT	83.2	88.5	94.9	98.8	100.9	124.8	132.8	142.5	154.1	162.2	65.3	57.5	50.7	37.8	23.8
MPJAT	83.9	87.5	92.6	98.0	100.9	125.7	134.3	143.0	154.6	163.4	65.6	57.7	51.7	41.4	32.8

Table 5.8: **Prediction Performance on 3DPW (Multi-Person Samples).** With respect to PA-MPJPE, the ZeroMotion baseline predicts more accurate poses in the short term and MPJAT predicts more accurate poses in the long-term. Similarly, we can see that when considering MPJPE, that JAT has the best long-term performance. With PCK both the ZeroMotion baseline and MPJAT are comparable up until 30 frames into the future where ZeroMotion performs significantly better than the rest of the models.

		3DPW - Prediction (Multi-Person Samples)														
		PA-MPJPE ↓					MPJPE ↓					PCK ↑				
Arch.	P_t	1	5	10	20	30	1	5	10	20	30	1	5	10	20	30
JAT	-	83.2	88.5	94.9	98.8	100.9	124.8	132.8	142.5	154.1	162.2	65.3	57.5	50.7	37.8	23.8
JAT	✓	81.0	84.9	89.7	96.0	99.7	124.8	131.5	140.1	152.9	161.7	63.2	55.4	48.9	36.6	24.2
MPJAT	-	83.4	87.6	93.2	97.7	100.4	125.0	132.6	141.6	153.7	163.0	65.8	58.0	51.3	40.2	27.5
MPJAT	✓	83.9	87.5	92.6	98.0	100.9	125.7	134.3	143.0	154.6	163.4	65.6	57.7	51.7	41.4	32.8

Table 5.9: **Ablation on P_t for Prediction (Multi-Person Samples).** JAT combined with the interpenetration loss P_t achieves the lowest PA-MPJPE and MPJPE of all transformer models, but it also leads to a reduction in PCK. Both MPJAT models with and without P_t have comparable PA-MPJPE/MPJPE and score higher on PCK than the JAT models, but including the interpenetration loss leads to best PCK performance.

We again run the ablation on P_t to see how the interpenetration loss affects the prediction results. We can see in **Table 5.9** that adding P_t to JAT begets a lower PA-MPJPE and MPJPE, but also results in a worse PCK measure (except for specifically 30 frames into the future). Conversely, removing P_t from the MPJAT model results in a lower PA-MPJPE and MPJPE but also reduces the longer-term PCK scores. Both with or without P_t, MPJAT scores better on PCK than the JAT models; when comparing the two MPJAT models against each other it can be seen that using the P_t loss increases the PCK score at 30 frames into the future by more than 5%. From this we can surmise that pairing the Agent-Aware attention mechanism with the interpenetration loss leads to the best long-term performance in PCK.

3DPW - Conditioning (All Samples)			
Arch.	PA-MPJPE ↓	MPJPE ↓	PCK ↑
PHD+T	88.9	131.0	66.9
JAT	95.8	140.4	65.4
MPJAT	94.9	141.1	65.5

Table 5.10: **Conditioning Performance on 3DPW (All Samples).** When evaluating on all samples of our 3DPW test split, PHD+T has the best conditioning performance with there being very little difference between JAT and MPJAT.

	3DPW - Prediction (All Samples)														
	PA-MPJPE ↓					MPJPE ↓					PCK ↑				
Arch.	1	5	10	20	30	1	5	10	20	30	1	5	10	20	30
ZeroMot	90.3	96.8	104.5	112.7	118.3	133.9	145.1	159.3	179.3	193.0	65.6	56.9	47.8	36.7	30.4
PHD+T	90.6	98.7	109.2	120.8	125.7	134.1	146.2	162.8	187.6	203.5	65.5	55.5	43.7	30.8	24.5
JAT	93.6	101.0	109.3	114.9	118.2	139.0	151.1	165.0	182.3	196.0	64.0	53.4	44.8	32.0	19.0
MPJAT	94.2	100.6	107.1	114.0	117.9	141.8	155.2	167.8	185.1	199.6	63.0	51.8	44.2	33.1	25.3

Table 5.11: **Prediction Performance on 3DPW (All Samples).** When evaluating for prediction for all samples of our 3DPW test split, we see that PHD+T shows better performance for all metrics in near-term prediction with MPJAT still achieving the best PA-MPJPE and PCK in the long-term.

5.4 Discussion

In this chapter we presented the quantitative experiments we undertook to evaluate the models we are proposing. We started by outlining details about the training process, we then described the datasets we used for our evaluation, and finally provided the results we collected on the performance of our models and the ablative experiments we conducted to measure the contributions of our additions.

Our results show that there is merit to both the Joint-Aware Transformer and the Multi-Person Joint-Aware Transformer. Specifically, we can see that when evaluated on a large dataset with uncomplicated scenes like Human3.6M, our JAT model can predict more accurate future poses than both the ZeroMotion baseline and the PHD+T model when measured by PA-MPJPE and MPJPE. JAT additionally offers the best trade-off between long-term and short-term performance on PCK for the Human3.6M dataset.

Similarly, when we strictly look at the multi-person samples of our 3DPW test set, we see that our MPJAT model achieves the best PA-MPJPE scores while performing almost as well as the ZeroMotion baseline at long-term prediction when measured by PCK. Further, our ablation experiment in **Table 5.9** shows that the both Agent-Aware attention and the interpenetration loss P_t are required to achieve good performance when measured by PCK. From this we can infer that our MPJAT model can implicitly model multi-person interactions to better predict future poses.

Considering all samples from our 3DPW test split however, paints a different picture. In **Table 5.10** we see that the PHD+T model has the best performance across all metrics on the conditioning task when evaluated on the entire 3DPW test split (both single-person and multi-person samples). In **Table 5.11** we show prediction results on the same split. Here we can see that in contrast to the multi-person only results in **Table 5.8**, when evaluated on all samples the ZeroMotion baseline performs best on all metrics.

We hypothesize that the better performance of the ZeroMotion baseline when including single-person samples in our 3DPW test set can be attributed to two main factors. First, the 3D metrics we use to evaluate pose (MPJPE and PA-MPJPE) present an incomplete picture. Both metrics compute the joint distances with the root joint located at the same position, meaning the pelvis position is ignored when calculating the error. This results in the predicted future translation of the meshes being ignored when the models are evaluated on these metrics. This benefits the ZeroMotion baseline as it does not predict future translation, it simply repeats the last observed mesh for all future predictions and has no movement around the scene. In future work it is worth investigating other metrics that take into account this translation when evaluating joint error.

The second factor is that the 3DPW dataset was captured using a dynamic, moving camera. Here, the camera operator continuously orients the camera such that the subject in the scene is always close to the center of the shot. By repeating meshes that exist close to the center of the frame, ZeroMotion results are likely to be performant on PCK.

Conversely, the models we propose predict translation in the camera coordinate frame. They must now implicitly predict camera motion along with subject motion due to the dynamic camera. This can quickly lead to compounding errors if the proposed models are incorrect about camera movement.

5.4.1 Failure Modes

In the course of evaluating the results of our experiments, we came across two primary failure modes that our models exhibit. The first, exemplified in **Figure D.5**, occurs when the input bounding boxes have significant overlap. In these cases multiple people exist within overlapping bounding boxes, and it can be ambiguous which bounding box corresponds to which person, leading to the models potentially reconstructing the same individual twice. The MPJAT model has shown some capability to better disentangle the individuals than the PHD+T and JAT models as seen in **Figure D.4**, but it still suffers from this issue. Recent approaches that focus on multi-person reconstruction [44, 14] have found that providing segmentation masks for each individual can alleviate this issue.

The second failure mode we came across is that the output predictions of all models' would occasionally be static or consist of small motion, as shown in **Figure D.3** and **Figure D.6**. This is most likely due to the fact that our predictions are deterministic, and as such the models are predicting a sequence that minimizes error among a diverse set of future possibilities [47]. Here, the best approach would be to switch to a stochastic prediction paradigm, but we leave that as future work.

Chapter 6

Conclusion

With this thesis we proposed an iterative set of modifications to the Predicting Human Dynamics (PHD) model that results in a method that can jointly predict the trajectory and pose of multiple people in a scene. We started by describing an approach to extend the original PHD model by predicting the trajectory of an individual. We dubbed this the PHD+T model, and treat it as our baseline. We then present our novel Joint-Aware Bias, an approach to inject egocentric information into a transformer's attention weights. Calling this model the Joint-Aware Transformer (JAT), we show that in unambiguous scenes like those in Human3.6M, this proposed approach is advantageous. Finally we introduced our Multi-Person Joint-Aware Transformer (MPJAT), an extension of the previous method that can simultaneously model multiple people. We show that when evaluated on samples consisting of multiple individuals, the MPJAT model outperforms repeated application of the PHD+T and JAT models on each individual independently. However, greater work is required to create a method that strongly outperforms the naive ZeroMotion baseline.

6.1 Limitations and Future Work

One key limitation that affects all of our proposed methods is the cropping step performed in the process of computing image features. While we can compute the spatial location of the individual in the scene using the bounding box coordinates, we lose the structure of the surrounding regions leading to a lack of global scene information. This can be addressed by implicitly modelling scene context following [9] or explicitly capturing interactions with objects in the environment as proposed in [97].

Another limitation is the deterministic prediction paradigm leveraged by our proposed methods. Given some input frames, the models will produce a single future outcome informed by the models' belief of what is most likely to happen next. Reality however does not guarantee that the future consists of only one possible trajectory and there are a multitude of ways that the future can unfurl from the same set of observations. Modelling this uncertainty is the next step in improving upon the models that we propose here. A stochastic approach can parameterize a distribution of possible futures and alleviate the regression-to-the-mean issue our work faces [47, 69, 70]. Further, in an online inference setting, stochastic models can be used to approximate prediction confidence; multiple future outcomes can be generated and poses from recently observed frames can be compared against these predictions to inform which future trajectories are more likely.

There exist other avenues of research that can also improve the applicability of our work. Examples include the introduction of a temporal discriminator to help ensure predictions of future meshes are consistent with respect to time (similar to the work done in [8]), swapping out the ResNet based backbone from HMR [21] with something that leverages a Vision Transformer like [15], or investigating approaches that better tackle issues of occlusion following [41, 98]. In addition to the items already addressed, training the models proposed here on a larger multitude of datasets with significantly more in the wild and multi-person samples has the potential to greatly improve performance.

6.2 Societal Impact

This thesis aims to propose a set of models that can directly infer future human motion based on data from a camera feed. Veritable success in this domain can have a far reaching affect on any technology that interacts with people in our tangible world; self-driving cars that can accurately predict pedestrian trajectories, industrial robotics that are built with proactive safety mechanisms, or domestic robots that can help the elderly and those in need of assistance around the house are all examples where the ability to forecast human behavior is crucial.

Like most technology, models that attempt to predict human behavior can just as much be a curse on society as they can be a boon. In an increasingly connected world, mass surveillance is becoming an ubiquitous fact of life. Large camera networks that leverage this technology to monitor a populace can have potentially life threatening outcomes for a community. Autocratic regimes or heavy handed agencies of well meaning governments can use the ideas discussed here to monitor for behavior that may progress into actions they deem unsavory; to say nothing of the privacy concerns this raises, a system that incorrectly predicts unwanted conduct can result in an innocent individual becoming a ceaseless fixation of authority. Even more hair-raising is the idea of automated weapon systems that can accurately model an individual's motion. While killer robots á la Terminator seem far-fetched, drone warfare is a very much a reality.

Addressing these considerations is a difficult task. Due to the relatively open nature of machine learning research most of the knowledge and technology required to build these systems is freely available. At this stage the only approach to prevent abuse is legislation. However much like internet legislation, threading the needle between over and under legislation of AI research is a balancing act that must be approached with nuance. Restricting research into the weaponization of AI in a similar vein as the Biological Weapons Convention, makes sense. Imposing limits on AI research as a whole however is questionable. Current machine learning research is already skewing towards groups with

large enough funds to throw data and computing resources at a problem, leaving smaller companies and academic groups to focus on niche problems or develop solutions with more modest resource requirements. Further constraining this by granting authority to conduct all AI research to a limited number of outfits can compound issues regarding bias in data and algorithms due to the stifling of new perspectives that can arise from having many sets of eyes on the same problem.

Bibliography

[1] "Amazon Warehouse Robots 'Increase Staff Injuries'," *BBC*, September 30, 2020. [Online]. Available: https://www.bbc.com/news/technology-54355803 [Accessed Dec. 10, 2023].

[2] S. D. Couto, "Robotic Arm Crushes Man to Death in South Korean Vegetable Packing Factory," *Global News*, November 9, 2023. [Online]. Available: https://globalnews.ca/news/10081179/man-crushed-to-death-robot-south-korea-factory/ [Accessed Dec. 10, 2023].

[3] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "ByteTrack: Multi-Object Tracking by Associating Every Detection Box," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[4] T. Meinhardt, A. Kirillov, L. Leal-Taixe, and C. Feichtenhofer, "TrackFormer: Multi-Object Tracking with Transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[5] D. Pavllo, C. Feichtenhofer, D. Grangier, and M. Auli, "3D Human Pose Estimation in Video with Temporal Convolutions and Semi-Supervised Training," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[6] Z. Tang, Z. Qiu, Y. Hao, R. Hong, and T. Yao, "3D Human Pose Estimation With Spatio-Temporal Criss-Cross Attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[7] A. Kanazawa, J. Y. Zhang, P. Felsen, and J. Malik, "Learning 3D Human Dynamics from Video," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5614–5623, 2019.

[8] M. Kocabas, N. Athanasiou, and M. J. Black, "VIBE: Video Inference for Human Body Pose and Shape Estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[9] Z. Cao, H. Gao, K. Mangalam, Q. Cai, M. Vo, and J. Malik, "Long-term Human Motion Prediction with Scene Context," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[10] X. Peng, S. Mao, and Z. Wu, "Trajectory-Aware Body Interaction Transformer for Multi-Person Pose Forecasting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[11] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, no. 5, p. 4282, 1995.

[12] J. Y. Zhang, P. Felsen, A. Kanazawa, and J. Malik, "Predicting 3D Human Dynamics from Video," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7114–7123, 2019.

[13] K. Xie, T. Wang, U. Iqbal, Y. Guo, S. Fidler, and F. Shkurti, "Physics-Based Human Motion Estimation and Synthesis From Videos," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[14] J. Rajasegaran, G. Pavlakos, A. Kanazawa, and J. Malik, "Tracking People by Predicting 3D Appearance, Location & Pose," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[15] S. Goel, G. Pavlakos, J. Rajasegaran, A. Kanazawa, and J. Malik, "Humans in 4D: Reconstructing and Tracking Humans with Transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[16] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A Generative Model for Raw Audio," in *Proceedings 9th ISCA Workshop on Speech Synthesis (SSW 9)*, p. 135, 2016.

[17] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal Convolutional Networks for Action Segmentation and Detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 156–165, 2017.

[18] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language Models are Few-Shot Learners," in *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020.

[19] W. Jiang, N. Kolotouros, G. Pavlakos, X. Zhou, and K. Daniilidis, "Coherent Reconstruction of Multiple Humans from a Single Image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[20] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "SMPL: A Skinned Multi-Person Linear Model," *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, vol. 34, pp. 248:1–248:16, Oct. 2015.

[21] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik, "End-to-End Recovery of Human Shape and Pose," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[22] Y. Huang, F. Bogo, C. Lassner, A. Kanazawa, P. V. Gehler, J. Romero, I. Akhter, and M. J. Black, "Towards Accurate Marker-Less Human Shape and Pose Estimation over Time," in *Proceedings of the International Conference on 3D Vision (3DV)*, 2017.

[23] A. Arnab, C. Doersch, and A. Zisserman, "Exploiting Temporal Context for 3D Human Pose Estimation in the Wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[24] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black, "Keep it SMPL: Automatic Estimation of 3D Human Pose and Shape from a Single Image," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[25] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. Gehler, and B. Schiele, "Deepcut: Joint subset partition and labeling for multi person pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[26] C. Lassner, J. Romero, M. Kiefel, F. Bogo, M. J. Black, and P. V. Gehler, "Unite the People: Closing the Loop Between 3D and 2D Human Representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4704–4713, 2017.

[27] G. Pavlakos, L. Zhu, X. Zhou, and K. Daniilidis, "Learning to Estimate 3D Human Pose and Shape from a Single Color Image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 459–468, 2018.

[28] N. Kolotouros, G. Pavlakos, M. J. Black, and K. Daniilidis, "Learning to Reconstruct 3D Human Pose and Shape via Model-fitting in the Loop," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[29] N. Kolotouros, G. Pavlakos, D. Jayaraman, and K. Daniilidis, "Probabilistic Modeling for Human Mesh Recovery," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[30] L. Müller, A. A. A. Osman, S. Tang, C.-H. P. Huang, and M. J. Black, "On Self-Contact and Human Pose," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[31] S. Tripathi, L. Müller, C.-H. P. Huang, O. Taheri, M. Black, and D. Tzionas, "3D Human Pose Estimation via Intuitive Physics," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[32] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[33] J. Zhang, D. Yu, J. H. Liew, X. Nie, and J. Feng, "Body Meshes as Points," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[34] Y. Sun, Q. Bao, W. Liu, Y. Fu, B. Michael J., and T. Mei, "Monocular, One-stage, Regression of Multiple 3D People," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[35] Y. Sun, W. Liu, Q. Bao, Y. Fu, T. Mei, and M. J. Black, "Putting People in their Place: Monocular Regression of 3D People in Depth," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[36] H. Rhodin, N. Robertini, D. Casas, C. Richardt, H.-P. Seidel, and C. Theobalt, "General Automatic Human Shape and Motion Capture using Volumetric Contour Cues," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[37] H. Choi, G. Moon, J. Y. Chang, and K. M. Lee, "Beyond Static Features for Temporally Consistent 3D Human Pose and Shape from a Video," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[38] E. Gärtner, M. Andriluka, E. Coumans, and C. Sminchisescu, "Differentiable Dynamics for Articulated 3D Human Motion Reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[39] E. Gärtner, M. Andriluka, H. Xu, and C. Sminchisescu, "Trajectory Optimization for Physics-Based Reconstruction of 3D Human Pose from Monocular Video," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[40] J. Li, S. Bian, C. Xu, G. Liu, G. Yu, and C. Lu, "D&D: Learning Human Dynamics from Dynamic Camera," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[41] Y. Yuan, U. Iqbal, P. Molchanov, K. Kitani, and J. Kautz, "GLAMR: Global Occlusion-Aware Human Mesh Recovery With Dynamic Cameras," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[42] V. Ye, G. Pavlakos, J. Malik, and A. Kanazawa, "Decoupling Human and Camera Motion from Videos in the Wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[43] Y. Sun, Q. Bao, W. Liu, T. Mei, and M. J. Black, "TRACE: 5D Temporal Regression of Avatars with Dynamic Cameras in 3D Environments," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[44] J. Rajasegaran, G. Pavlakos, A. Kanazawa, and J. Malik, "Tracking People with 3D Representations," in *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

[45] Z. Qiu, Q. Yang, J. Wang, H. Feng, J. Han, E. Ding, C. Xu, D. Fu, and J. Wang, "PSVT: End-to-End Multi-Person 3D Pose and Shape Estimation With Progressive Video Transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[46] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-End Object Detection with Transformers," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 213–229, 2020.

[47] E. Barsoum, J. Kender, and Z. Liu, "HP-GAN: Probabilistic 3D Human Motion Prediction via GAN," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017.

[48] K. Fragkiadaki, S. Levine, P. Felsen, and J. Malik, "Recurrent Network Models for Human Dynamics," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015.

[49] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-RNN: Deep Learning on Spatio-Temporal Graphs," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[50] J. Martinez, M. J. Black, and J. Romero, "On Human Motion Prediction Using Recurrent Neural Networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[51] R. J. Williams and D. Zipser, "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, 1989.

[52] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks," in *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, p. 1171–1179, 2015.

[53] M. Wei, L. Miaomiao, S. Mathieu, and L. Hongdong, "Learning Trajectory Dependencies for Human Motion Prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[54] M. Wei, L. Miaomiao, and S. Mathieu, "History Repeats Itself: Human Motion Prediction via Motion Attention," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[55] W. Mao, M. Liu, M. Salzmann, and H. Li, "Multi-Level Motion Attention for Human Motion Prediction," *International Journal of Computer Vision*, vol. 129, no. 9, pp. 2513–2535, 2021.

[56] M. Li, S. Chen, Z. Zhang, L. Xie, Q. Tian, and Y. Zhang, "Skeleton-Parted Graph Scattering Networks for 3D Human Motion Prediction," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[57] W. Guo, Y. Du, X. Shen, V. Lepetit, X. Alameda-Pineda, and F. Moreno-Noguer, "Back to MLP: A Simple Baseline for Human Motion Prediction," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023.

[58] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[59] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[60] H. Zhao and R. P. Wildes, "Where are you Heading? Dynamic Trajectory Prediction with Expert Goal Examples," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[61] V. Adeli, M. Ehsanpour, I. D. Reid, J. C. Niebles, S. Savarese, E. Adeli, and H. Rezatofighi, "TRiPOD: Human Trajectory and Pose Dynamics Forecasting in the Wild," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[62] J. Wang, H. Xu, M. Narasimhan, and X. Wang, "Multi-Person 3D Motion Prediction with Multi-Range Transformers," in *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

[63] W. Guo, X. Bie, X. Alameda-Pineda, and F. Moreno-Noguer, "Multi-Person Extreme Motion Prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[64] A. Vemula, K. Muelling, and J. Oh, "Social Attention: Modeling Attention in Human Crowds," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[65] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.

[66] Y. Yuan, X. Weng, Y. Ou, and K. Kitani, "AgentFormer: Agent-Aware Transformers for Socio-Temporal Multi-Agent Forecasting," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[67] J. Walker, K. Marino, A. K. Gupta, and M. Hebert, "The Pose Knows: Video Forecasting by Generating Pose Futures," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[68] A. Hernandez, J. Gall, and F. Moreno, "Human Motion Prediction via Spatio-Temporal Inpainting," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[69] Y. Yuan and K. Kitani, "DLow: Diversifying Latent Flows for Diverse Human Motion Prediction," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[70] Y. Zhang, M. J. Black, and S. Tang, "We are More than Our Joints: Predicting how 3D Bodies Move," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[71] R. Li, S. Yang, D. A. Ross, and A. Kanazawa, "AI Choreographer: Music Conditioned 3D Dance Generation with AIST++," 2021.

[72] W. Mao, M. Liu, and M. Salzmann, "Weakly-supervised Action Transition Learning for Stochastic Human Motion Prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[73] M. Petrovich, M. J. Black, and G. Varol, "Action-Conditioned 3D Human Motion Synthesis with Transformer VAE," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[74] T. Lucas*, F. Baradel*, P. Weinzaepfel, and G. Rogez, "PoseGPT: Quantization-based 3D Human Motion Generation and Forecasting," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[75] M. Hassan, D. Ceylan, R. Villegas, J. Saito, J. Yang, Y. Zhou, and M. Black, "Stochastic Scene-Aware Motion Prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[76] Y. Zheng, Y. Yang, K. Mo, J. Li, T. Yu, Y. Liu, K. Liu, and L. J. Guibas, "GIMO: Gaze-Informed Human Motion Prediction in Context," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[77] J. Carreira, P. Agrawal, K. Fragkiadaki, and J. Malik, "Human Pose Estimation with Iterative Error Feedback," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4733–4742, 2016.

[78] M. Loper, N. Mahmood, and M. J. Black, "MoSh: Motion and Shape Capture from Sparse Markers," *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, vol. 33, pp. 220:1–220:13, Nov. 2014.

[79] Z. Li, J. Liu, Z. Zhang, S. Xu, and Y. Yan, "CLIFF: Carrying Location Information in Full Frames into Human Pose and Shape Estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[80] N. Kolotouros, G. Pavlakos, and K. Daniilidis, "Convolutional Mesh Regression for Single-Image Human Shape Reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[81] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the Continuity of Rotation Representations in Neural Networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5738–5746, 2019.

[82] H. Tung, A. W. Harley, W. Seto, and K. Fragkiadaki, " Adversarial Inverse Graphics Networks: Learning 2D-to-3D Lifting and Image-to-Image Translation from Unpaired Supervision ," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[83] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least Squares Generative Adversarial Networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2794–2802, 2017.

[84] P. J. Liu, M. Saleh, E. Pot, B. Goodrich, R. Sepassi, L. Kaiser, and N. Shazeer, "Generating Wikipedia by Summarizing Long Sequences," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[85] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, "Improving Language Understanding by Generative Pre-Training," tech. rep., OpenAI, 2018.

[86] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," tech. rep., OpenAI, 2019.

[87] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-Attention with Relative Position Representations," in *Proceedings of the North American Chapter of the Association for Computational Linguistics*, 2018.

[88] D. Hendrycks and K. Gimpel, "Gaussian Error Linear Units (GELUs)," *arXiv preprint arXiv:1606.08415*, 2016.

[89] X. Ren and X. Wang, "Look Outside the Room: Synthesizing A Consistent Long-Term 3D Scene Video from A Single Image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[90] C. M. Jiang, A. Cornman, C. E. Park, B. Sapp, Y. Zhou, and D. Anguelov, "MotionDiffuser: Controllable Multi-Agent Motion Prediction using Diffusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[91] K. Arora, L. El Asri, H. Bahuleyan, and J. Cheung, "Why Exposure Bias Matters: An Imitation Learning Perspective of Error Accumulation in Language Generation," in *Findings of the Association for Computational Linguistics (ACL)*, 2022.

[92] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, pp. 1325–1339, jul 2014.

[93] T. von Marcard, R. Henschel, M. Black, B. Rosenhahn, and G. Pons-Moll, "Recovering Accurate 3D Human Pose in The Wild Using IMUs and a Moving Camera," in *Proceedings of the European Conference on Computer Vision (ECCV)*, sep 2018.

[94] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[95] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization," *Journal of Machine Learning Research*, vol. 18, p. 6765–6816, Jan. 2017.

[96] J. C. Gower, "Generalized Procrustes Analysis," *Psychometrika*, vol. 40, pp. 33–51, 1975.

[97] J. Y. Zhang, S. Pepose, H. Joo, D. Ramanan, J. Malik, and A. Kanazawa, "Perceiving 3D Human-Object Spatial Arrangements from a Single Image in the Wild," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[98] M. Kocabas, C.-H. P. Huang, O. Hilliges, and M. J. Black, "PARE: Part Attention Regressor for 3D Human Body Estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[99] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd ed., 2004.

Appendix A

Full Perspective Projection

Full perspective projection maps a 3D point in camera coordinate space to a 2D point on the image plane, where the image plane represents the picture captured by a camera [99]. Perspective projection can be performed by simply right-multiplying the camera's intrinsic matrix $\boldsymbol{K} \in \mathbb{R}^{3\times3}$ with a 3D point of interest $\boldsymbol{P} \in \mathbb{R}^3$. The intrinsic camera matrix captures the internal properties of the camera where

$$\boldsymbol{K} = \begin{bmatrix} F_x & 0 & \frac{W}{2} \\ 0 & F_y & \frac{H}{2} \\ 0 & 0 & 1 \end{bmatrix}, \tag{A.1}$$

consists of the focal length in the x and y dimension, as well as the principal offset $\left(\frac{W}{2},\ \frac{H}{2}\right)$ which encodes where the principal point lies on the image plane. Given the intrinsic camera matrix, we can project a 3D point $\boldsymbol{P}$ with

$$\lambda \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix} = \begin{bmatrix} F_x & 0 & \frac{W}{2} \\ 0 & F_y & \frac{H}{2} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{A.2}$$

where $\boldsymbol{p} \in \mathbb{R}^2 = [p_x\ p_y]^\top$ represents the 2D location of the point on the image plane.

Appendix B

6D Rotation Representation

Given a rotation matrix $\boldsymbol{M} \in SO(3)$ where

$$\boldsymbol{M} = \begin{bmatrix} | & | & | \\ \mathbf{m}_1 & \mathbf{m}_2 & \mathbf{m}_3 \\ | & | & | \end{bmatrix}, \tag{B.1}$$

the 6D rotation representation $\boldsymbol{R}$ is computed by simply dropping the last column $\boldsymbol{m}_3$, i.e.,

$$\boldsymbol{R} = \begin{bmatrix} | & | \\ \mathbf{m}_1 & \mathbf{m}_2 \\ | & | \end{bmatrix}. \tag{B.2}$$

To compute the rotation matrix $\boldsymbol{M}$ given some 6D representation $\boldsymbol{R}$ the process is

$$\mathbf{R} = \begin{bmatrix} | & | \\ \mathbf{m}_1 & \mathbf{m}_2 \\ | & | \end{bmatrix},$$

$$\begin{aligned} \mathbf{b}_1 &= N(\boldsymbol{m}_1), \\ \mathbf{b}_2 &= N(\boldsymbol{m}_2 - (\boldsymbol{b}_1 \cdot \boldsymbol{m}_2)\boldsymbol{b}_1), \\ \mathbf{b}_3 &= \boldsymbol{b}_1 \times \boldsymbol{b}_2, \end{aligned} \tag{B.3}$$

$$\mathbf{M} = \begin{bmatrix} | & | & | \\ \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 \\ | & | & | \end{bmatrix}.$$

Here $\times$ refers to the cross product and

$$N(\boldsymbol{x}) = \frac{\boldsymbol{x}}{\|\boldsymbol{x}\|_2}. \tag{B.4}$$

Appendix C

Hyperparameters

C.1 PHD+T

PHD+T Hyperparameters		
	Φ Num Channels	2048
	Γ Num Channels	128
f_{movie}	Kernel Size	3
	Num Temporal Convolution Blocks	3
	Num Group Norm Groups	16
	Receptive Field	13
f_{AR}	Kernel Size	3
	Num Temporal Convolution Blocks	3
	Num Group Norm Groups	17
	Receptive Field	13

Table C.1: **PHD+T Model Hyperparameters.**

PHD+T Learning Rates (Conditioning)	
LR Schedule	None
f_{movie}	2.0×10^{-5}
f_{HMR}	2.0×10^{-5}
discriminator	2.0×10^{-4}

Table C.2: **PHD+T Learning Rates for Conditioning.**

PHD+T Learning Rates (Prediction)	
LR Schedule	None
f_{AR}	2.3×10^{-6}
f_{JE}	6.8×10^{-4}
discriminator	1.9×10^{-3}

Table C.3: **PHD+T Learning Rates for Prediction.**

C.2 JAT/MPJAT

JAT/MPJAT Hyperparameters		
	$\mathbf{\Phi}$ Num Channels	512
	$\mathbf{\Gamma}$ Num Channels	128
f_{movie}	Num Layers	1
	Num Attention Heads	1
	Max Receptive Field	43
f_{AR}	Num Layers	1
	Num Attention Heads	1
	Max Receptive Field	46

Table C.4: **JAT/MPJAT Model Hyperparameters.**

JAT/MPJAT Learning Rates (Conditioning)		
LR Schedule	Warmup Type	Linear
	Initial LR Ratio	0.31
	Num Warmup Epochs	44
	Decay Type	Cosine
	Num Decay Epochs	13
	Final LR Ratio	0.06
Peak LR	f_{movie}	6.1×10^{-4}
	f_{HMR}	1.0×10^{-4}
	Discriminator	3.2×10^{-4}

Table C.5: **JAT/MPJAT Learning Rates for Conditioning.**

JAT/MPJAT Learning Rates (Prediction)		
LR Schedule	Warmup Type	Linear
	Initial LR Ratio	0.16
	Num Warmup Epochs	21
	Decay Type	Cosine
	Num Decay Epochs	8
	Final LR Ratio	0.07
Peak LR	f_{AR}	2.4×10^{-5}
	f_{JE}	3.1×10^{-5}
	Discriminator	2.0×10^{-3}

Table C.6: **JAT/MPJAT Learning Rates for Prediction.**

Appendix D

Model Outputs

D.1 Human3.6M Samples

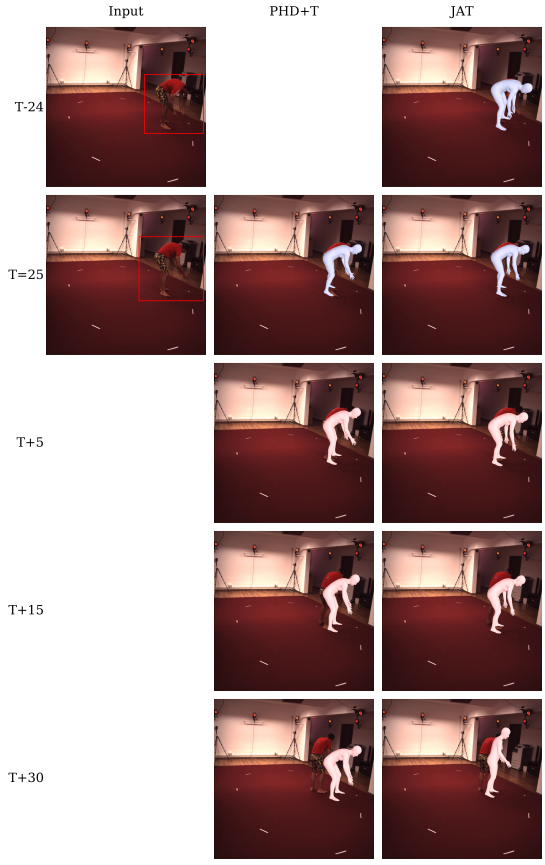


Figure D.1: **Visualization of Model Predictions on Human3.6M Sample 1.**

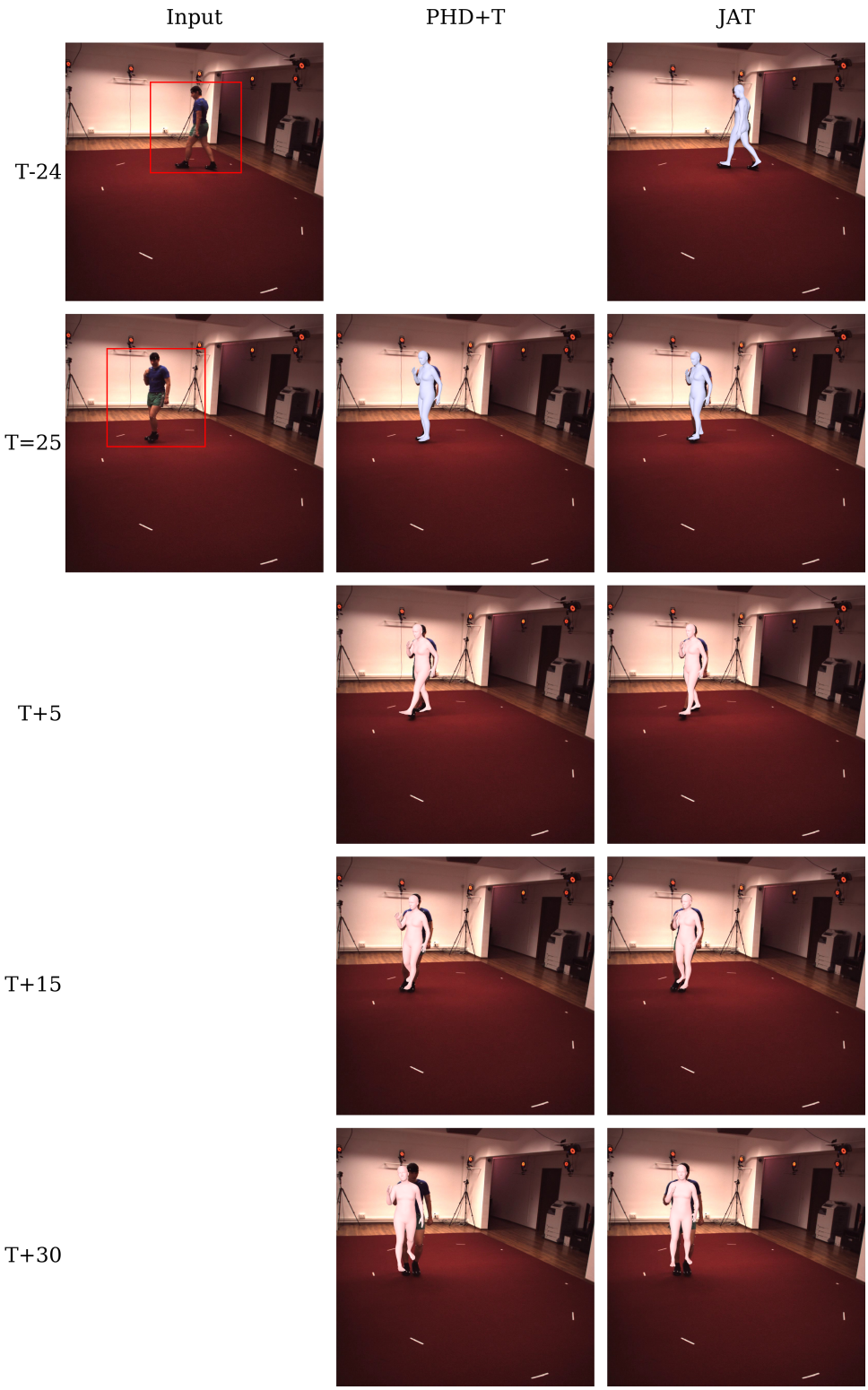


Figure D.2: **Visualization of Model Predictions on Human3.6M Sample 2.**

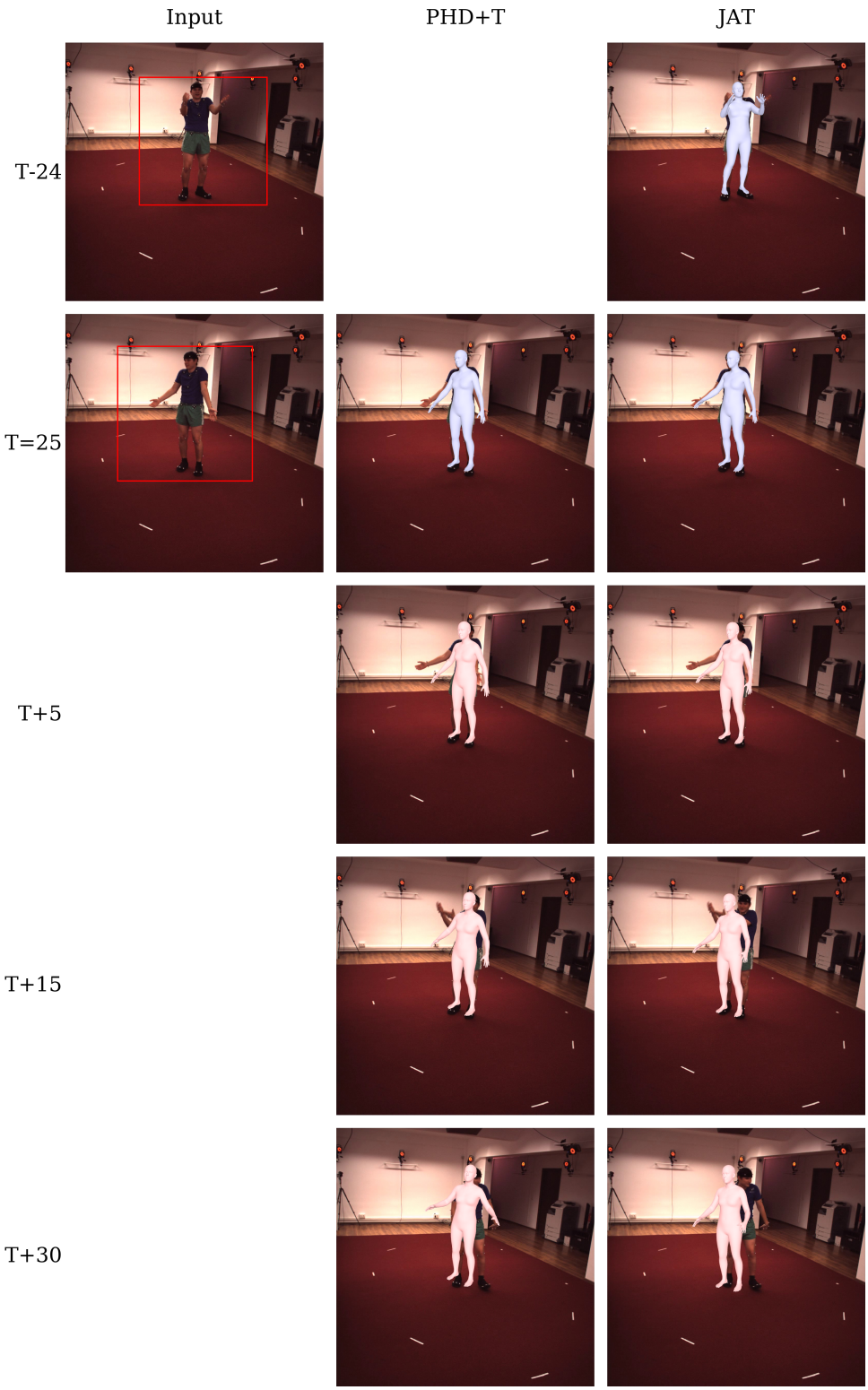


Figure D.3: **Visualization of Model Predictions on Human3.6M Sample 3.**

D.2 3DPW Samples

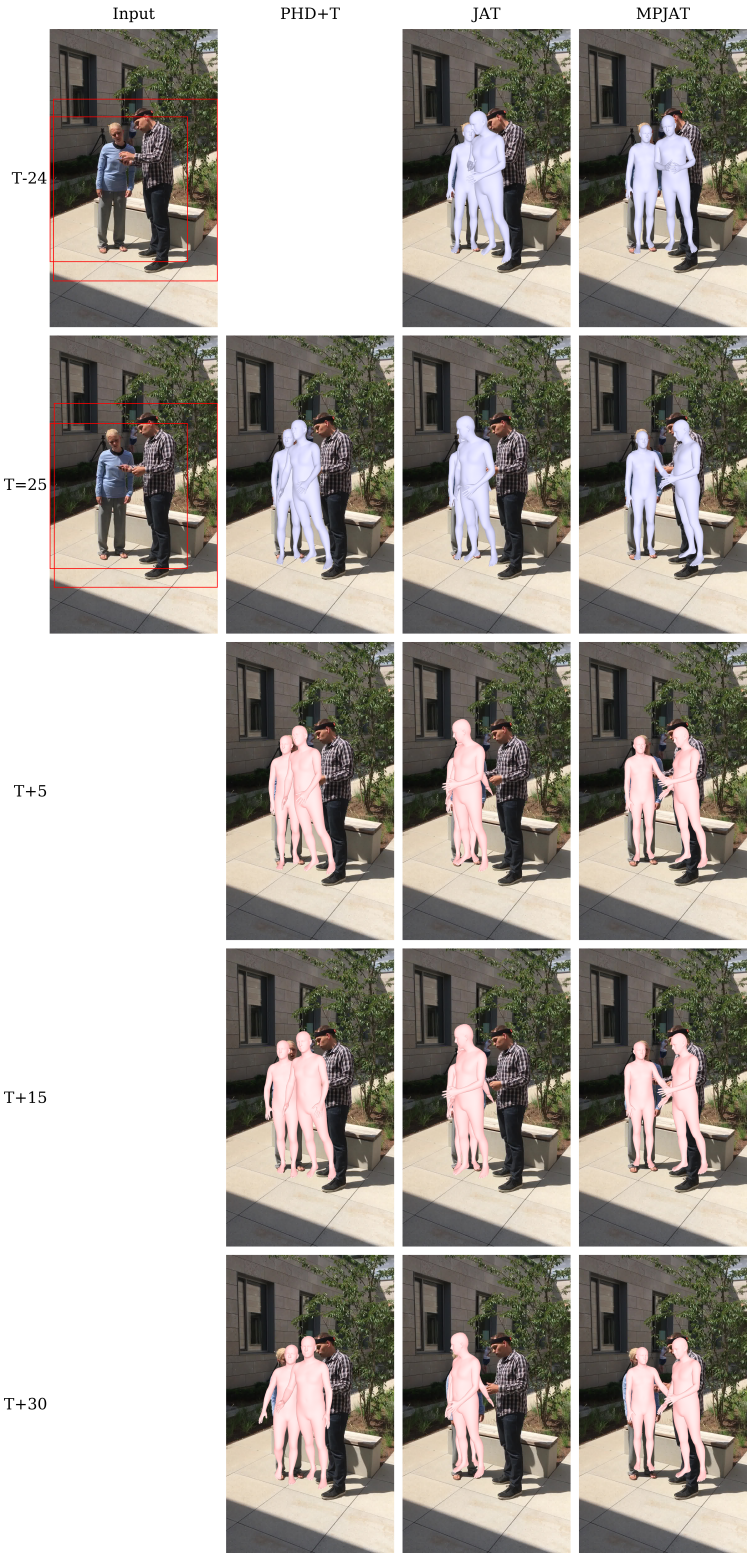


Figure D.4: **Visualization of Model Predictions on 3DPW Sample 1.**

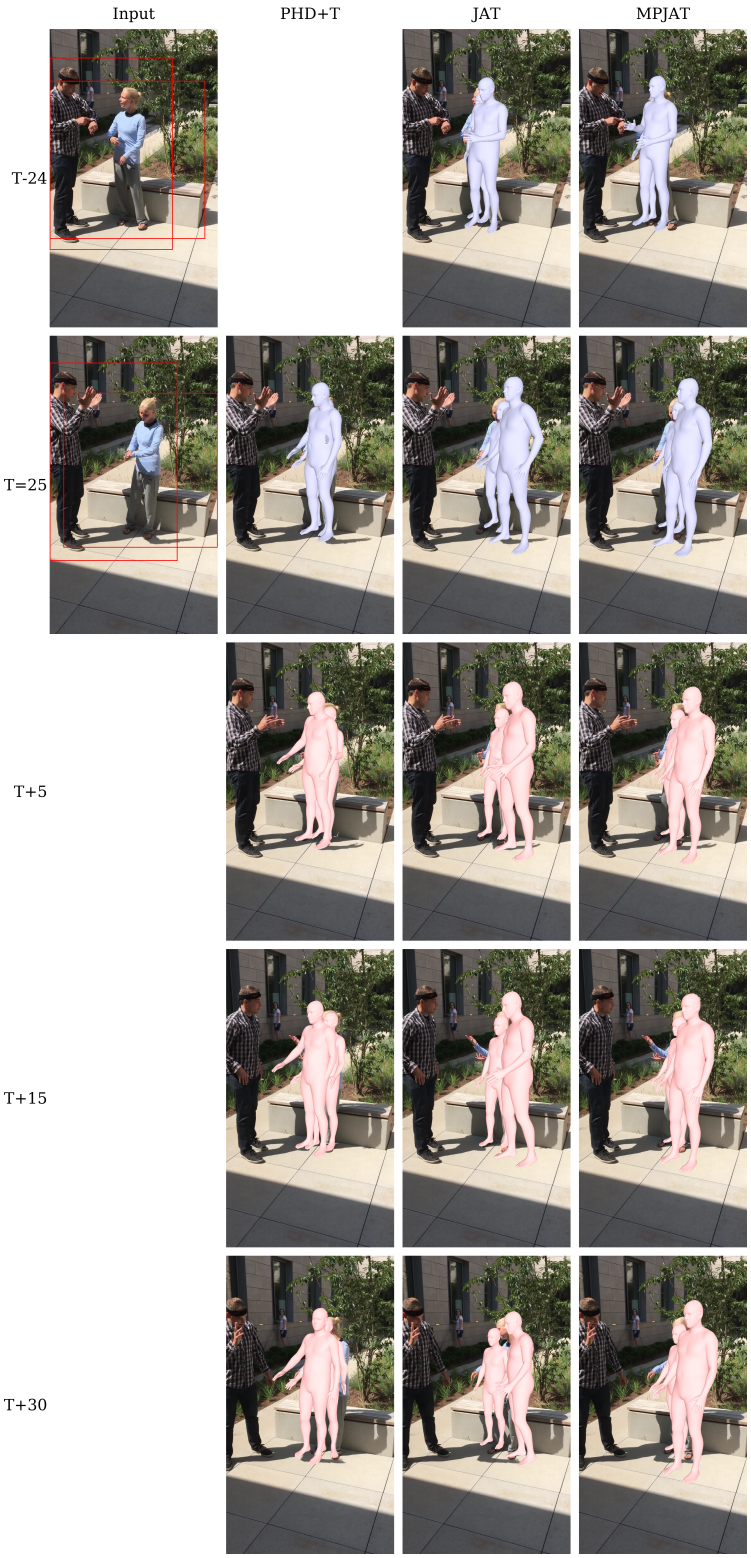


Figure D.5: **Visualization of Model Predictions on 3DPW Sample 2.**

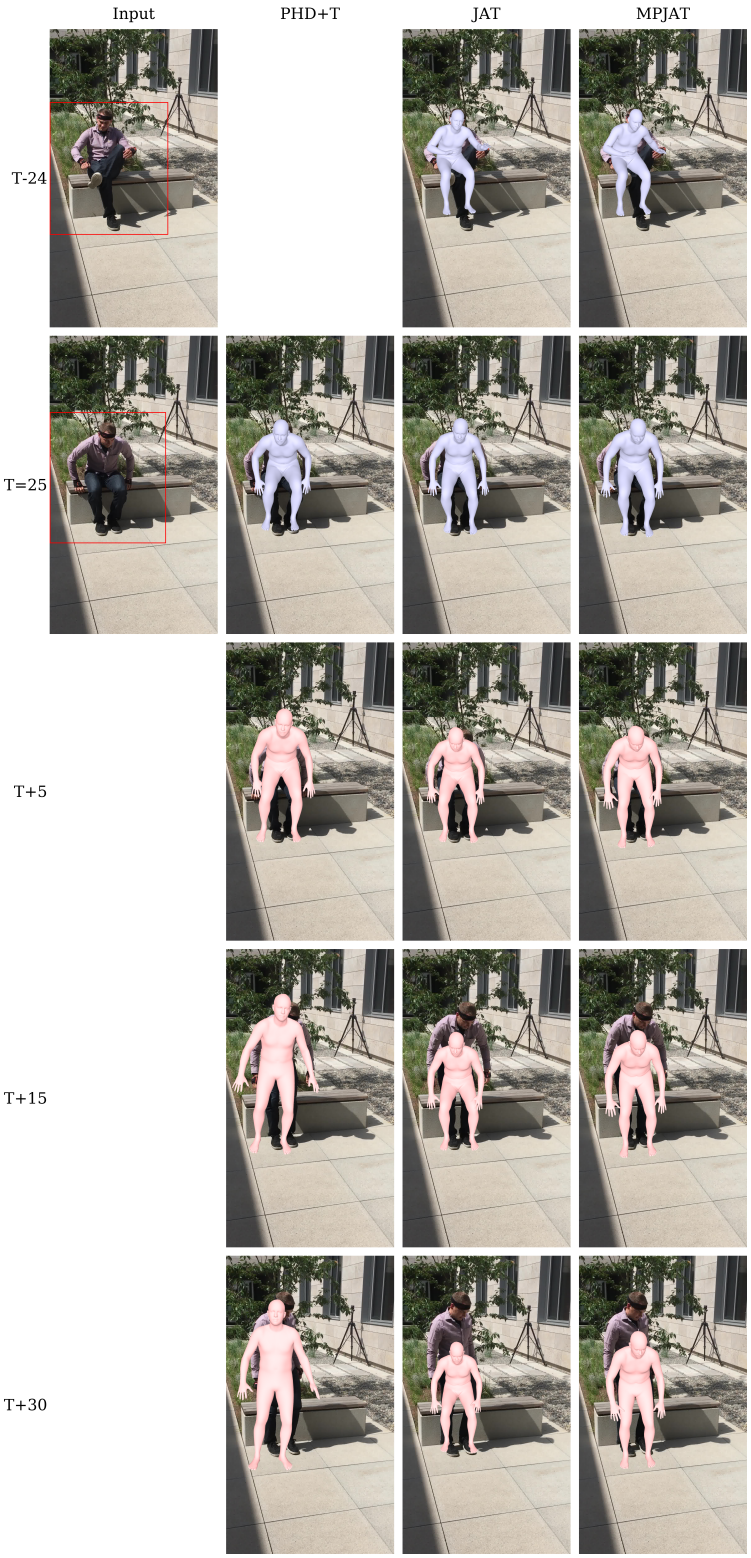


Figure D.6: **Visualization of Model Predictions on 3DPW Sample 3**