Neural Localizer Fields for Continuous 3D Human Pose and Shape Estimation

István Sárándi[1,2] **Gerard Pons-Moll**[1,2,3]
[1]University of Tübingen, Germany, [2]Tübingen AI Center, Germany,
[3]Max Planck Institute for Informatics, Saarland Informatics Campus, Germany
https://istvansarandi.com/nlf

Abstract

With the explosive growth of available training data, single-image 3D human modeling is ahead of a transition to a data-centric paradigm. A key to successfully exploiting data scale is to design flexible models that can be supervised from various heterogeneous data sources produced by different researchers or vendors. To this end, we propose a simple yet powerful paradigm for seamlessly unifying different human pose and shape-related tasks and datasets. Our formulation is centered on the ability – both at training and test time – to query any arbitrary point of the human volume, and obtain its estimated location in 3D. We achieve this by learning a continuous neural field of body point localizer functions, each of which is a differently parameterized 3D heatmap-based convolutional point localizer (detector). For generating parametric output, we propose an efficient post-processing step for fitting SMPL-family body models to nonparametric joint and vertex predictions. With this approach, we can naturally exploit differently annotated data sources including mesh, 2D/3D skeleton and dense pose, without having to convert between them, and thereby train large-scale 3D human mesh and skeleton estimation models that considerably outperform the state-of-the-art on several public benchmarks including 3DPW, EMDB, EHF, SSP-3D and AGORA.

1 Introduction

Along with the wider field of computer vision, the human pose and shape estimation community has recently started a transition towards leveraging large-scale training data to produce robust models [93, 30, 12, 83, 60]. When assembling large training sets from different sources, it is inevitable that they will use different types of annotations, including different parametric mesh models, 3D skeletal joints and markers from different motion capture vendors and manual 2D keypoint annotations. Re-annotation to a single format for training is a costly and error-prone process, and therefore difficult to scale. Similarly, current models are trained to output a specific format, such as pre-defined joints, keypoints, or parametric body meshes, which is limiting for downstream applications which require other formats. We therefore argue that a good model design should be able to ingest and output highly heterogeneous forms of annotations in a unified manner.

Despite the different kinds of labels, these tasks all share a common underlying factor: the correspondence problem of mapping from a canonical human body space to the observation space of the camera. Hence, if a model could localize arbitrary human points (both surface and body-internal ones, e.g., joints), its predictions could be supervised from any human-centric data source with point labels and could be customized to new landmark sets even after deployment. In this work, we propose a method that is able to achieve exactly this.

To localize individual points, we leverage the machinery of volumetric heatmap estimation with soft-argmax decoding, which has worked well in skeletal keypoint estimation methods [86, 104, 92]. However, generalizing this to arbitrary points is nontrivial. In prior work, each body joint heatmap is

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

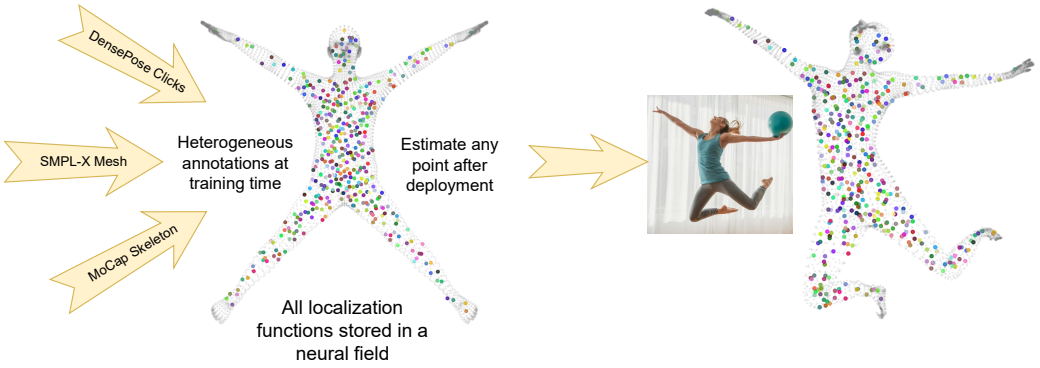

Figure 1: *Can one model learn to localize any point of the human body in 3D from a single RGB image?* We propose to build a generalist human pose and shape estimator that can readily learn from any annotated points at training time and can estimate any user-chosen points at test time.

produced through a separate set of convolution weights, tying the model to a particular set of joints or keypoints. Extending this paradigm to localize an infinite number of points would require infinite memory to store the weights, and learning would be highly inefficient as weights would need to be trained independently for each point.

Our key idea and contribution is to learn a continuous field of point localizers, which we call a *Neural Localizer Field*. The principle is to modulate a deep network's prediction layer on-the-fly, based on what point we need to predict: at training time querying points for which we have annotations, and at test time sampling the points that an application requires. Here, we are inspired by the recent success of neural field representations, such as neural radiance [75], distance [84, 18] and occupancy fields [74], but instead of predicting a physical property (occupancy, color) we predict a function.

Specifically, this *localizer field*'s input domain is the full 3D volume of the canonical human body and its output range is the parameter space of the modulated convolutional output layer (making the localizer field a hypernetwork [33, 46, 13]). By optimizing the neural field's parameters, we obtain a smooth field of localizers (functions) over the human volume domain, improving robustness, and knowledge sharing. In the paper, we investigate various design choices related to this architecture, including positional encodings. This architecture results in consistent and accurate skeletal joint predictions along with smooth nonparametric mesh output, whose format is chosen by the user at test time, see Fig. 1. Since a parametric representation (for example based on SMPL [65]) is often preferred for further downstream processing, as the second contribution of this paper, we derive a simple and efficient algorithm for fitting SMPL-family body models to our non-parametric predictions. This involves alternating between estimating global body part orientations through the Kabsch algorithm [44], and solving for the shape coefficients in a regularized linear least squares formulation. This converges after as few as ca. 2–4 rounds, and is well-suited for GPU acceleration.

In summary, our main contribution is the *Neural Localizer Field* method for the 3D localization of an *infinite continuum of points within the human body volume*, based on a single RGB image, allowing seamless mixed-dataset training using various skeleton or mesh annotation formats without the need for tedious and error-prone re-annotation. Through this formulation, we are able to train a generalist human pose and shape estimator that outperforms prior work on a wide range of important benchmarks. We will make our code and trained models publicly available for research.

2 Related Work

Methods for 3D pose/shape estimation use a wide variety of formats: different skeletons, meshes, body models or volumetric representations. For a comprehensive review, see [108]. Here we focus on methods that closely relate to ours.

Large-scale multi-dataset human-centric training. A few recent works [93, 30, 12] have started to explore scaling up 3D human body reconstruction methods beyond the previously common practice of training on one or a few datasets. Models trained at scale from many sources tend to outperform specialized models that were trained for a specific dataset. However, current methods have to unify

their datasets to a common format [12], which is often an ill-posed problem. This can introduce errors if done through pseudo-annotation, requiring significant human effort and manual quality checks. Another option is to train a model to predict all possible outputs and optimize consistency losses between them [93], but this is not scalable to a large, possibly infinite number of annotation types. By stark contrast, our model can ingest data from many different formats, and can be asked to output any point set at test time.

Nonparametric mesh estimation and sparse keypoints. Some works [53, 68] predict further keypoints on the surface beyond the skeleton, noting that this carries richer shape information. Ma et al. [68] predict a sparse set of virtual markers and approximate the rest through convex interpolation. Nonparametric mesh estimation has been a successful line of recent research [19, 122, 62, 63, 22, 20, 52, 76, 67], mainly due to better preservation of alignment in contrast to parametric estimation, which often suffers from mean-shape bias [96]. In contrast to our work, the prominent Transformer-based approaches of this category [62, 19, 63] feed a static set of keypoints into their architecture throughout training, which then exchange information through self-attention layers, and require a coarse to fine strategy and complex design for efficient processing. In contrast, our architecture is more lightweight, and can directly produce predictions for all possible points, without having to interpolate them from some other set of points.

Continuous and dense representations. While the dominant paradigm to supervise pose estimators has been sparse keypoints and skeletons, a few works have explored dense, continuously varying keypoints for supervision and prediction. DenseReg [1] predicts a dense 2D deformation grid to self-supervise a face regressor. In DensePose [31, 79], 2D keypoint estimation is generalized to a continuous representation, where arbitrary surface points can be regression targets, instead of having a fixed set of sparse keypoints throughout the dataset. DensePose3D [100] shows how 2D dense maps can be used to lift estimates to 3D. Similar in spirit, DenseBody [120, 127] predicts 3D mesh points on a 2D UV-map. Chandran et al. [15] estimate 2D facial landmarks in a continuous manner. Common to all these works is that either the prediction or some intermediate representations are given in 2D, e.g., UV-maps, dense semantic maps or 2D output coordinates. By contrast, we are able to perform a direct 3D-to-3D mapping from canonical to observation space, to localize an infinite continuum of points in the human volume, to harness heterogeneous annotations.

Distinctions from DensePose. DensePose [31, 79] may seem similar enough to our motivation that we should expand upon why it does not solve the same problem we are tackling. In a sense, our problem formulation is the *reverse* of DensePose's. DensePose starts out in image space and asks, *per pixel*, what canonical surface point is visible at the pixel (facing the camera). This inherently restricts the formulation to the frontal surface of the body. Occluded points, body-internal joints and the opposite-side surface of the human are ignored in DensePose. Instead, we start in canonical space and ask the more natural question of where each body point (identified in canonical space) can be found in 3D observation space. This is more in line with the goals in practical applications. However, because of this reverse relation between the two formulations, the annotations in DensePose datasets [78, 31] can still be directly used to supervise our model as well (with the roles reversed), further boosting the data sources available to us.

3D keypoint estimation through heatmaps. A well-established paradigm to predict 3D keypoints is to use 3D heatmaps [86, 104, 92, 68, 73, 71]. The advantage is that convolutional architectures are well suited for localizing joints in the image, and inputs and outputs are well aligned. However, predicting an infinite number of heatmaps for a continuum of points is not possible, as it would require storing an infinite number of last-layer weights. With our Neural Localizer Fields, we can harness the well-performing heatmap paradigm and adapt it to predicting keypoints chosen at test time.

Parametric fitting to nonparametric predictions. Fitting a parametric mesh to keypoint predictions is needed for certain applications and for compactness. If correspondences are known, the classical way is to solve a nonlinear least squares problem as in the original SMPL [65] paper, but this requires many iterations and is slow. To speed up and to make the process differentiable, some works train an MLP to predict body parameters from sparse markers [52, 126]. This requires generalizing to all possible poses which is hard. By contrast, we propose a solver based on limb rotation estimation via the Kabsch algorithm and linear least-squares shape fitting. This direct geometry-based algorithm is orders of magnitude faster than a classical optimization procedure, and does not suffer from generalization issues of the MLP methods since it is not learning-based.

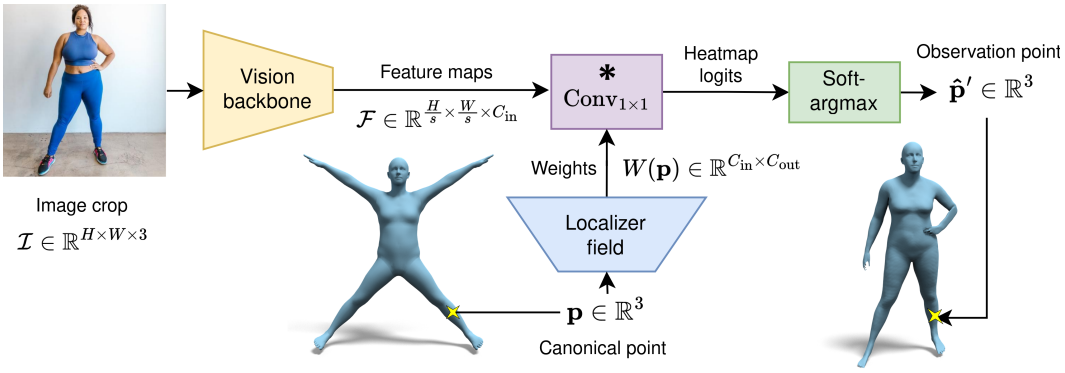

Figure 2: **Overview of NLF.** Given image features $\mathcal{F}$ and any arbitrarily chosen 3D query point $\mathbf{p}$ within the canonical human volume, we aim to estimate the observation-space 3D point $\mathbf{p}'$. To control which point gets estimated, we dynamically modulate a convolutional layer at the output, to produce heatmaps for the requested point. We achieve this modulation by predicting the convolutional weights through a neural field. During training, the points $\mathbf{p}$ can be picked per training example based on whichever points are annotated for it, allowing natural dataset mixing. At test time, the model can flexibly estimate any surface point and any skeletal joint inside the body volume, as required.

3 Method

Our method consists of three main parts: (1) a *point localizer network* (PLN) for localizing 3D points from images, (2) a *neural localizer field* (NLF) to control what point the PLN should predict, and (3) an efficient body model fitting algorithm to create a parametric representation for a set of points that were localized nonparametrically. We metonymically refer to our complete approach as *NLF*. This modular structure establishes clear interfaces and can enable independent improvements on the parts. The architecture overview is depicted in Fig. 2.

3.1 Point Localizer Network

We start with describing the basic mechanism of localizing a point in our architecture, which we then generalize to arbitrary points afterward.

Given an RGB image crop $\mathcal{I} \in [0,1]^{H \times W \times 3}$ of a person, we first extract feature maps $\mathcal{F} \in \mathbb{R}^{h \times w \times C}$ through an off-the-shelf backbone at stride $s = W/w = H/h$. We then follow the MeTRAbs [92, 93] architecture in skeletal 3D pose estimation, and produce metric-scale truncation-robust volumetric heatmaps $H_{\text{3D}} \in \mathbb{R}^{h \times w \times D}$, as well as 2D pixel-space heatmaps $H_{\text{2D}} \in \mathbb{R}^{h \times w}$ through a 1×1 convolutional layer. The heatmaps are decoded to coordinates via soft-argmax [54, 80, 104], and the resulting 2D and root-relative 3D points are combined to an absolute camera-space estimate, including estimates for truncated body parts. Additionally, we predict an uncertainty map $U \in \mathbb{R}^{h \times w}$, which will be averaged to an uncertainty scalar per point using weights from H_{2D}.

A classical, joint-based pose estimator would be trained to produce a fixed set of heatmaps, as many as there are joints in the skeleton, and it would have to be trained and tested with the corresponding fixed set of joints. Taking a functional view, observe that this type of architecture realizes a discrete set of **localizer functions**, mapping image features to 3D locations as

$$f_i : \mathbb{R}^{h \times w \times C} \to \mathbb{R}^3, \quad \mathcal{F} \mapsto \hat{\mathbf{p}}'_i, \quad i \in [1..J] \tag{1}$$

with J being the number of joints the model can predict. In a pose estimator, each f_i mapping is represented through a section of the weight matrix W of the convolutional output head. Hence, in this paradigm, implementing localizer functions to detect further points requires extending the weight matrix, giving linear growth in the parameters of the output layer. It is clearly infeasible to store the weights for every possible point we might want to predict. Furthermore, even though the functions for localizing nearby points of the body have more in common than those detecting faraway points, this spatial structure is lost when representing them as discrete outputs. In the next part, we describe our method for retrieving the full continuum of localizer functions, allowing predictions for any point, as well as ensuring spatial information sharing across the functions.

3.2 Neural Localizer Field

The goal of our work is to go beyond predicting only a predetermined finite set of localizable points in a network, and to instead learn to localize *all* points of the body with one universal model, by learning a whole *localizer field* of functions Ψ. The field Ψ associates a function $f_{\mathbf{p}}$ to every point $\mathbf{p}$ in the canonical human volume $\Omega_H \subset \mathbb{R}^3$:

$$\Psi : \Omega_H \to \left(\mathbb{R}^{h\times w\times C} \to \mathbb{R}^3\right), \quad \mathbf{p} \mapsto (f_{\mathbf{p}} : \mathcal{F} \mapsto \mathbf{p}'), \tag{2}$$

where the posed point $\mathbf{p}'$ is determined based on image features $\mathcal{F}$ by $f_{\mathbf{p}}$. Although neural fields are typically used to predict points or vectors, here we use them to predict localizer functions $f_{\mathbf{p}}$. This way, we represent information on a continuous domain, while appropriately modeling both smooth structure and fine-grained details of a signal. Specifically, $f_{\mathbf{p}}(\mathcal{F})$ is implemented as a single-layer convolutional head whose weights are predicted as a function $w(\mathbf{p})$ of the canonical point $\mathbf{p}$ we want to localize. The function $w : \Omega_H \to \mathbb{R}^{(C+1)\cdot(D+2)}$ is realized as an MLP, taking three canonical coordinates of $\mathbf{p}$ as input and producing parameters $(\mathbf{W}, \mathbf{b})$ for $f_{\mathbf{p}}$'s convolutional layer to control which point gets predicted (cf. hypernetworks). Here the $C+1$ is the backbone feature dimensionality plus one bias, and $D+2$ are the total number of output channels which are D depth channels of the volumetric heatmap H_{3D}, one 2D heatmap H_{2D} and one uncertainty map U. In summary, given the image features $\mathcal{F}$ and a canonical query point $\mathbf{p}$, the following computation takes place:

$$(\mathbf{W}, \mathbf{b}) := w(\mathbf{p}) \tag{3}$$

$$(H_{\text{2D}}, H_{\text{3D}}, U) := \text{conv}_{1\times1}(\mathcal{F}; \mathbf{W}, \mathbf{b}) \tag{4}$$

$$\mathbf{p}'_{\text{3D, rootrel}} := \text{soft-argmax}(H_{\text{3D}}) \qquad \mathbf{p}'_{\text{2D}} := \text{soft-argmax}(H_{\text{2D}}) \tag{5}$$

$$u := \sum\nolimits_{x,y} U_{x,y}\, \text{softmax}(H_{\text{2D}})_{x,y} \quad \sigma := \text{softplus}(u) + \epsilon, \tag{6}$$

after which the 3D root-relative and the 2D predictions are fused into a camera-space output $\mathbf{p}'$ following [92]. The localizer field's architecture consists of a positional encoding part and an MLP with GELU activation (see the appendix for the details of the MLP layers).

Positional encoding. To express high-frequency signals, neural fields typically employ positional encodings [90, 107]. Since the field's domain is the canonical human volume Ω_H, we use the eigenbasis ϕ_i of the volumetric Laplacian Δ (the Fourier basis equivalent for bounded volumes), inspired by the use of the Laplace–Beltrami operator in [79]. We then compute the *global point signature* (GPS) [91]

$$\gamma(\mathbf{p}) = \left[\phi_1(\mathbf{p})/\sqrt{\lambda_1}, ..., \phi_M(\mathbf{p})/\sqrt{\lambda_M}\right]^T \in \mathbb{R}^M, \tag{7}$$

a widely used descriptor in shape matching. So far, this only gives us the eigenfunction values ϕ_i on the discrete tetrahedral mesh nodes. To obtain an approximate interpolant for the entire domain, we distill the eigenfunctions into a small learnable Fourier feature network [58] $h : \mathbb{R}^3 \to \mathbb{R}^M$ with $h_i(\mathbf{p}) \approx \phi_i(\mathbf{p})$. The network h can be finetuned end-to-end. Details are in the appendix. We will compare the effectiveness of plain coordinates, global point signature, and learnable Fourier features.

Mathematical formulation for training. Our formulation allows us to supervise the point-localization task on a continuous domain. Given an image $\mathcal{I}$ with features $\mathcal{F}$, and the ground-truth continuous warp function $g : \Omega_H \to \mathbb{R}^3$ mapping between the canonical and observation spaces, the loss measures the distance between ground truth $g(\cdot)$ and prediction $f(\cdot; \mathcal{F})$ in function space, through the volume integral

$$\mathcal{L} = \iiint_{\Omega_H} \mathcal{L}_{\text{p}}\big(f(\mathbf{p}; \mathcal{F}), g(\mathbf{p})\big)\, \mathrm{d}^3\mathbf{p}, \tag{8}$$

where $\mathcal{L}_{\text{p}}(\cdot, \cdot)$ is the pointwise loss comparing prediction $f(\mathbf{p}; \mathcal{F}) = \hat{\mathbf{p}}'$ to ground truth $g(\mathbf{p}) = \mathbf{p}'$ in observation space. We adopt the Euclidean loss

$$\mathcal{L}_{\text{p}}(\hat{\mathbf{p}}', \mathbf{p}') = \|\hat{\mathbf{p}}' - \mathbf{p}'\| \tag{9}$$

without squaring, for better outlier-robustness. (In the appendix, we show how the uncertainty output can be combined into the loss function, as this aspect is not the main focus here.) This loss is applied to the predictions in the 3D camera coordinate frame, the 3D root-relative frame and the 2D image

space as well, with weighting factors. Naturally, only the 2D loss is applied for training examples that only have 2D labels.

In practice, we often do not have the full volume annotation $g(\mathbf{p})$ available for all $\mathbf{p}$ as in (8), and it would be too expensive to evaluate the integral for every image. Since we train on a mixture of datasets, each training example $\left(\mathcal{I}_i, \{\mathbf{p}'_k\}_{k=1}^{K_i}\right)$ can contain a different annotated point set $\{\mathbf{p}'_k\}_{k=1}^{K_i}$ depending on the dataset the example comes from. Hence, we turn the integral (8) into a discrete sum over the available dataset-specific locations $\mathbf{p}_k \in \Omega_H$. For training examples with parametric body annotations, we perform Monte Carlo approximation of the integral by random sampling of points.

Zoo of annotations. Data in HPE contains a wide variety of annotation types including parametric meshes, 3D keypoints, 2D keypoints and DensePose samples. Thanks to our formulation described above, we can directly consume this data, since the zoo of annotations consists of subsets of points in the canonical volume. For examples that are annotated with a parametric mesh model, we sample 640 points uniformly over the surface and 384 points from the interior of the human volume. The sampled points are unique to each example of the batch. We supervise interior points densely to enable the network to predict new joint sets anywhere in the volume at test time. To deform interior points, we use scattered data interpolation [102, 9] to propagate SMPL deformations to the volume. For datasets annotated with 2D or 3D keypoints, the corresponding canonical positions need to be established per joint. We use an approximate initialization per skeleton type (see appendix), and finetune the precise canonical positions during the main training process. In case of DensePose annotations, we map the I, U, V labels to the corresponding SMPL surface points in the canonical pose.

Implementation details. We use EfficientNetV2-S (256 px) and L (384 px) [106] initialized from [93], and train with AdamW [66], linear warmup and exponential learning rate decay for 300k steps. Training the S model takes $\sim$2 days on two 40 GB A100 GPUs, while the L takes $\sim$4 days on 8 A100s.

3.3 Efficient Body Model Fitting in Post-Processing

Above, we introduced our method to nonparametrically localize any human point from an image, in 3D. While nonparametric prediction has its advantages (e.g., closer fits to the image data), in some cases a parametric representation (joint rotations θ and shape vector β) is preferable due to its compactness, and disentanglement of pose and shape. We therefore develop a fast algorithm to solve for pose and shape parameters that closely reproduce our predicted nonparametric vertices.

Our algorithm alternates between two steps for a small number of iterations ($\sim$3), followed by an adjustment step. We first independently fit global orientations per body part by applying a weighted Kabsch algorithm. Then, keeping the orientations fixed, we solve for β and the translation vector via linear least squares, since SMPL's vertex positions are linear in the shape parameters. Finally, we refine the part rotations in one pass along the kinematic tree, anchoring the rotations at the parent joint. For improved efficiency, it is possible to consider only a subset of the vertices when fitting. Further, given that our model estimates per-point uncertainties, these can be used in a weighted variant of the above algorithm. Splitting the shape estimation task into a nonparametric point localization and a fitting step has the further benefit that it is easy to share the body shape parameters β for multiple observations of the same person (multi-view and temporal use cases) during the least-squares regression. More details and the pseudocode of the algorithm are given in the appendix.

4 Experiments

We extensively evaluate our method on a variety of benchmarks: 3DPW [114] and EMDB [47] for SMPL body, AGORA [85] and EHF [87] for SMPL-X, SSP-3D [96] for SMPL focusing on body shape, as well as Human3.6M [40], MPI-INF-3DHP [70] and MuPoTS-3D [72] for 3D skeletons.

Our goal is to enable large-scale multi-dataset training from heterogeneous annotation sources in order to train strong models. To show that our formulation is effective at this, we assemble a large meta-dataset for use in training. Due to space constraints we provide the detailed dataset description in the appendix. For parametric meshes we include the datasets [85, 8, 6, 35, 50, 109, 24, 36, 16, 130, 121, 11, 119, 123, 88, 10, 112], with points sampled directly from each dataset's native body model (one of SMPL, SMPL-X or SMPL-H, neutral or gendered, as provided) without a need for any conversions. For 3D skeleton annotations, we use [40, 70, 72, 124, 131, 43, 57, 81, 115, 89,

Table 1: **Ablation of positional encodings.** Input representation matters in NLF. The fixed Laplacian-derived global point signature helps most metrics, but is outperformed by learnable Fourier features. Best results are achieved by initializing the learnable Fourier features to the GPS.

	SSP3D mIoU↑	3DPW P-MVE↓	EMDB P-MVE↓	3DPW MPJAE↓
Plain XYZ coordinates	0.777	54.0	57.7	16.9
Global point signature (GPS)	0.789	54.2	57.5	15.8
Learnable Fourier features	0.804	**52.8**	57.0	15.4
Learnable Fourier (GPS init.)	**0.812**	**52.8**	**56.4**	**15.3**

34, 29, 132, 111, 82, 110, 23, 5, 28, 26, 27, 4, 48, 17]. Several of these have custom skeletons that are nontrivial to convert to the SMPL family, preventing existing 3D HPS methods from exploiting these rich data sources. For 2D keypoint annotations we use [37, 41, 113, 2, 116, 25] and further use DensePose datasets [78, 31] which provide pairs of SMPL surface points and corresponding pixel locations. Integrating all these datasets would be a great challenge if we had to choose one specific representation. Instead, we can easily train using the strategy described in Sec. 3.2, we simply map each annotation convention to the canonical human volume and learn a continuous localizer field.

We use standard metrics including per-joint error (MPJPE), per-vertex error (MVE), orientation angle error (MPJAE) and their Procrustes variants ("P-"). Metrics are explained in detail in the appendix.

Pose prediction. NLF outperforms all baselines on **EMDB** (Tab. 3), which has challenging outdoor sequences. The improvement is drastic, we get an MPJPE of 68.4 mm compared to the second-best (98.0 mm). Remarkably, we outperform the temporal method WHAM (79.7 mm) although we work frame-by-frame. **3DPW** results confirm the same (Tab. 2). (For reference, the standard error of the mean under normal assumptions for 3DPW MPJPE is 0.06 mm.) We attribute this performance to the fact that we can natively train on many sources without error-prone conversions. (For better comparability to temporal methods, we also provide NLF results using a (non-learned) 5-frame temporal smoothing filter.) On **AGORA** (Tab. 5), we outperform all prior works in SMPL-X prediction. Even though we do not specially target facial and hand keypoints, we obtain the second best scores for these. We also achieve state-of-the-art body results on **EHF** (Tab. 7). The higher hand error is due to few detailed hands in the training data. We obtain excellent performance on the skeleton estimation benchmarks **Human3.6M, MPI-INF-3DHP and MuPoTS-3D** as well (Tab. 6). This evaluation is easy with NLF, as we can pick at test time *which points of the body to predict*.

Shape prediction. Mesh recovery methods are often biased towards an average shape. Keypoint-based methods are pixel-aligned but ignore shape, and SMPL parametric regressors tend to produce subpar image alignment. By contrast, since we model the full continuum of points, we can estimate shape more accurately while being pixel-aligned. This is reflected on **SSP-3D** (Tab. 4), a dataset specifically designed to evaluate shape estimation, where we again improve over all baselines, obtaining a low error of 10.0 mm. Notably, we outperform works, such as ShapeBoost [7], which are specialized to shape estimation and have not been tested on general pose estimation benchmarks. Figure 3 shows a qualitative example. Following the protocol in [98], we also evaluate combined shape estimation from multiple (max. 5) images of the same person, by shared estimation of the β shape parameters during body model fitting. This reduces the error to 9.6 mm. Our NLF achieves state-of-the-art shape estimation while *simultaneously* obtaining SOTA results on the most challenging pose benchmarks such as EMDB, using the same model weights.

Positional encoding. To assess the impact of positional encoding on shape fidelity, we evaluate the mIoU silhouette overlap measure on the shape-diverse SSP-3D, the Procrustes-aligned vertex error on 3DPW and EMDB, as well as the orientation error on 3DPW after SMPL-fitting using the small backbone. As seen in Sec. 4, the use of the Laplacian-based global point signature (GPS) encoding leads to a better mIoU of 0.789 compared to the 0.777 achieved using plain X, Y, Z coordinates. We attribute this to better representation of fine-grained geometry. Learnable Fourier features obtain even better results than GPS, but best is the combination where we use the GPS as initialization for the learnable Fourier feature network. Do note however that NLF achieves good performance even with plain coordinates.

Uncertainty estimation. We provide an ablation for uncertainty predictions in the appendix.

Table 2: **Results on 3DPW (14 joints).** * denotes *temporal* (multi-frame) method

Method	Trained without 3DPW-train				Trained with 3DPW-train			
	MPJPE	P-MPJPE	MVE	P-MVE	MPJPE	P-MPJPE	MVE	P-MVE
PyMAF [128]	78.0	47.1	91.3		74.2	45.3	87.0	
SMPLer-X-H32 [12]	75.0	50.6			71.7	48.0		
BEDLAM-CLIFF [8]	72.0	46.6	85.0		66.9	43.0	78.5	
HybrIK [55]	71.6	41.8	82.3					
HMR 2.0a [30]	70.0	44.5						
Multi-HMR [3]	69.5	46.9	88.8		61.4	41.7	75.9	
*WHAM-B (ViT, w/ BEDLAM)** [101]					*56.9*	*35.7*	*67.4*	
NLF-S	60.9	38.5	73.3	52.8	55.6	35.9	67.0	48.9
NLF-S +fit	60.8	37.9	72.2	51.4	56.6	35.7	66.7	47.9
NLF-L	60.3	37.3	71.4	50.2	**54.1**	33.7	**63.7**	45.3
NLF-L +fit	**59.0**	**36.5**	**69.7**	**48.8**	54.9	**33.6**	**63.7**	**44.5**
NLF-L +fit +*smooth**	*57.2*	*35.4*	*67.8*	*47.7*	*53.2*	*32.6*	*62.1*	*43.5*

Table 3: **Results on EMDB1 (24j).** **temporal* method

Method	MPJPE ↓	P-MPJPE ↓	MVE ↓	P-MVE ↓	MPJAE ↓	P-MPJAE ↓
PyMAF [128]	131.1	82.9	160.0	98.1	28.5	25.7
LGD [103]	115.8	81.1	140.6	95.7	25.2	25.6
ROMP [105]	112.7	75.2	134.9	90.6	26.6	24.0
PARE [49]	113.9	72.2	133.2	85.4	24.7	22.4
GLAMR* [125]	107.8	71.0	128.2	85.5	25.5	23.5
FastMETRO-L [19]	115.0	72.7	133.6	86.0	25.1	22.9
CLIFF [59]	103.1	68.8	122.9	81.3	23.1	21.6
HybrIK [55]	103.0	65.6	122.2	80.4	24.5	23.1
HMR 2.0 [30]	98.0	60.6	120.3			
BEDLAM-CLIFF [8]	98.0	60.6	111.6			
*WHAM-B (ViT)** [101]	*79.7*	*50.4*	*94.4*			
NLF-S	74.5	44.9	87.8	56.4	–	–
NLF-S +fit	72.0	44.6	85.2	55.4	17.1	16.4
NLF-L	69.6	41.2	82.4	52.0	–	–
NLF-L +fit	**68.4**	**40.9**	**80.6**	**51.1**	**16.1**	**15.4**
NLF-L +fit +*smooth**	*66.7*	*39.9*	*78.7*	*50.0*	*16.0*	*15.2*

Table 4: **Results on SSP-3D.** PVE-T-SC is the per-vertex error in T-pose with scale correction, mIoU measures silhouette overlap. * means estimating body shape from multiple images of a person (max. 5).

Method	PVE-T-SC↓	mIoU↑
HMR [45]	22.9	0.69
SPIN [51]	22.2	0.70
SHAPY [21]	19.2	0.71
SoY [94]	15.8	0.76
STRAPS [96]	15.9	0.80
Sengupta et al. [98]	15.2	–
Sengupta et al. * [98]	*13.3*	–
Sengupta et al. [97]	13.6	–
LASOR [118]	14.5	0.67
HuManiFlow [99]	13.5	–
KBody [133]	25.6	0.80
ShapeBoost [7]	11.4	–
NLF-S +fit	**10.0**	**0.85**
NLF-S +fit (*shared* β)*	*9.6*	*0.85*
NLF-L +fit	11.1	0.83

Table 5: **Results on AGORA-test (SMPL-X).** Models are fine-tuned on AGORA.

Method	NMVE ↓		NMJE ↓		MVE ↓					MPJPE ↓				
	All	Body	All	Body	All	Body	Face	LHan	RHan	All	Body	Face	LHan	RHan
Hand4Whole [77]	144.1	96.0	141.1	92.7	135.5	90.2	41.6	46.3	48.1	132.6	87.1	46.1	44.3	46.2
BEDLAM [8]	142.2	102.1	141.0	101.8	103.8	74.5	**23.1**	**31.7**	**33.2**	102.9	74.3	**24.7**	**29.9**	**31.3**
PyMAF-X [129]	141.2	94.4	140.0	93.5	125.7	84.0	35.0	44.6	45.6	124.6	83.2	37.9	42.5	43.7
OSX [61]	130.6	85.3	127.6	83.3	122.8	80.2	36.2	45.4	46.1	119.9	78.3	37.9	43.0	43.9
HybrIK-X [56]	120.5	73.7	115.7	72.3	112.1	68.5	37.0	46.7	47.0	107.6	67.2	38.5	41.2	41.4
SMPLer-X [12]	107.2	68.3	104.1	66.3	99.7	63.5	29.9	39.1	39.5	96.8	61.7	31.4	36.7	37.2
Multi-HMR [3]	102.0	63.4	101.8	64.1	95.9	59.6	27.7	40.2	40.9	95.7	60.3	29.2	38.1	39.0
NLF-S	108.6	67.9	106.3	67.4	102.1	63.8	29.5	42.7	42.8	99.9	63.4	31.7	38.7	39.2
NLF-L	**98.6**	**62.1**	**96.6**	**61.9**	**92.7**	**58.4**	27.0	37.9	38.1	**90.8**	**58.2**	28.5	34.4	34.9

Table 6: **Results on skeleton estimation benchmarks.**

Method	Human3.6M		MPI-INF-3DHP			MuPoTS-3D
	MPJPE↓	P-MPJPE↓	PCK↑	AUC↑	MPJPE↓	PCK-detected↑
MeTRAbs-ACAE-S [93]	40.2	31.1	96.3	58.7	57.9	94.7
MeTRAbs-ACAE-L [93]	**36.5**	**27.8**	97.1	60.1	55.4	95.4
NLF-S	40.4	30.6	96.6	57.9	59.9	94.7
NLF-L	39.7	28.5	**97.5**	**61.0**	**54.9**	**95.5**

Table 7: **Results on EHF (SMPL-X).**

Method	MVE			P-MVE				P-MPJPE	
	Full-body	Hands	Face	Full-body	Body	Hands	Face	Body (14j)	Hands
HybrIK-X [56]	121.4	52.1	41.9	59.8	50.0	17.6	8.1	60.8	17.9
OSX [61]	70.8	53.7	26.4	48.7	–	15.9	6.0	–	–
PyMAF-X [129]	64.9	29.7	19.7	50.2	44.8	**10.2**	5.5	52.8	**10.3**
SMPLer-X [12]	62.4	47.1	17.0	37.1	–	14.1	**5.0**	–	–
Multi-HMR [3]	42.0	**28.9**	18.0	28.2	–	10.8	5.3	–	–
NLF-S	39.7	49.7	15.2	30.9	30.4	24.5	7.4	32.8	28.2
NLF-S +fit	40.0	49.7	15.4	30.4	29.7	24.0	6.5	32.0	25.4
NLF-L	**36.3**	43.2	**13.5**	**25.8**	24.8	21.3	6.7	26.3	25.3
NLF-L +fit	36.4	43.0	13.9	26.0	**24.4**	20.7	6.3	**26.1**	21.8

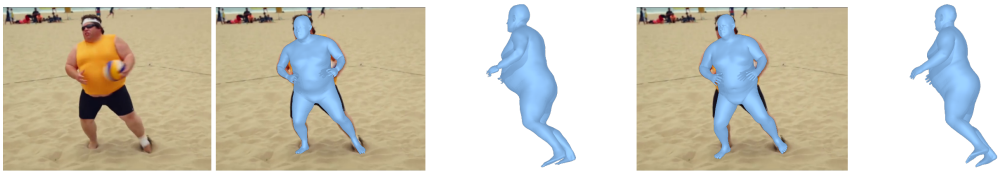

Figure 3: **Qualitative result on SSP-3D.** *left:* NLF's nonparametric output (front and side view), *right:* result of our proposed fast SMPL fitting algorithm (front and side). Our nonparametric prediction already has high quality, allowing us to use a simple and efficient fitting algorithm to obtain body model parameters that faithfully represent the nonparametric output.

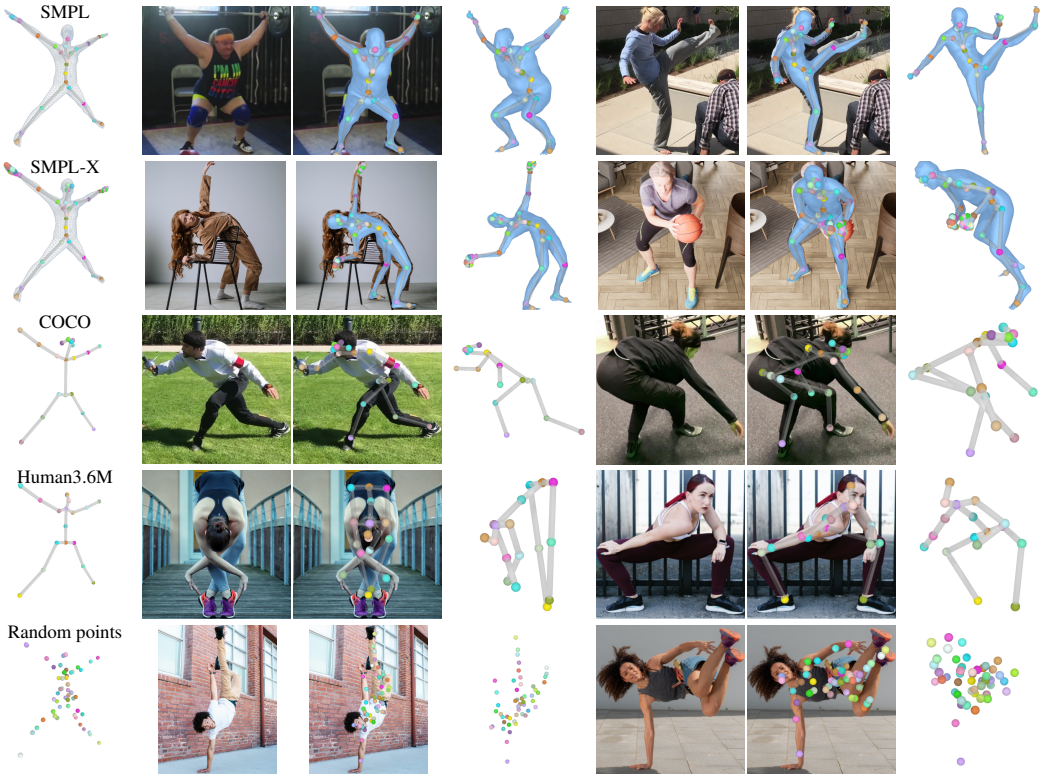

Figure 4: **Customizable point localization.** By selecting points $\mathbf{p}$ in the continous canonical space, we can predict any landmark set both at training and test time. The first column depicts the query points we estimate: SMPL(-X) joints and vertices, COCO joints, Human3.6M joints, and arbitrary points sampled within the human volume. The fourth and seventh column show rotated views.

Table 8: **Training data ablation.** Using all datasets yields the best results compared to only the real subset or only the synthetic subset, showing that NLF benefits from both kinds of data and that more data improves results. (Here we trained for 100k steps as opposed to 300k for the main evaluations.)

Method / data	SSP-3D PVE-T-SC↓	H36M MPJPE↓	3DHP PCK↑	3DPW P-MPJPE↓	3DPW P-MVE↓	EMDB P-MPJPE↓	EMDB P-MVE↓
NLF-S, real datasets only	11.9	43.0	96.0	40.4	55.5	47.8	59.3
NLF-S, synthetic datasets only	10.0	48.0	95.4	39.2	53.5	**45.7**	57.9
NLF-S, all datasets	**9.9**	**41.6**	**96.3**	**38.4**	**53.1**	45.8	**57.2**

Effect of datasets. Assessing the contribution of each individual dataset is computationally infeasible, nonetheless, to validate that NLF benefits from diverse supervision, we partition the data into real and synthetic subsets. As Tab. 8 shows, best results are achieved when all datasets are used in training.

Real-time inference. NLF-S has a batched throughput of 410 fps and unbatched throughput of 79 fps on an Nvidia RTX 3090 GPU. For NLF-L these are 109 fps and 41 fps respectively.

Qualitative results. To demonstrate the versatility of our method in localizing any point of the human body, we show qualitative results in Fig. 4 according to different representations. Qualitative results with the final fitted SMPL are given in the appendix.

Efficient body model fitting. On both 3DPW and EMDB, fitting SMPL to the nonparametric estimation (Sec. 3.3) generally reduces the error, but the difference is small, indicating that the original predictions already have high quality. To measure the fitting speed, we consider the task of fitting SMPL-X to vertices derived from SMPL (model transfer) using sample data from the SMPL-X website (33 meshes). The official code for the transfer takes 33 minutes with mean vertex error 5.0 mm. Our algorithm takes 28.4 ms with error 7.8 mm, i.e., orders of magnitude faster, and error increase is negligible relative to the typical prediction error of centimeters.

Limitations. NLF works frame-by-frame without temporal cues, and estimates each person independently, so highly-overlapping people remain challenging. Further, our predictions may have self-intersections and the absolute distance may be inaccurate due to depth/scale ambiguity.

Broader social impact. Human pose and shape estimation can help advance assistive technologies, human–robot interaction or interactive entertainment. However, like other vision methods, a potential for misuse exists in e.g., illegitimate surveillance. We will release models for research only.

5 Conclusion

We proposed *Neural Localizer Fields* to upgrade 3D human pose estimators from fixed joint sets to predicting any point of the body surface and volume. We store point-localizer functions in a neural field, as opposed to storing explicit convolutional weights for every joint the network can predict. This enables us to train a large-scale generalist pose and shape estimation model, without any effort on relabeling all data to one format. Instead, we can supervise our model with any points that happen to be annotated for particular training examples. Besides this training advantage, we can choose at runtime which points in the volume to predict, allowing us to output any desired format. To make our output even more useful in downstream applications, we proposed an efficient points-to-parameters fitting algorithm to obtain SMPL parameters. Our model trained on several datasets with different formats achieves state-of-the-art performance with significant improvement over baselines on several benchmarks. Our model exhibits good performance on outdoor, indoor and synthetic data, regardless of the skeleton/mesh format used for different benchmarks. With the recent trend of more datasets from different sources being available, NLF paves the way to leverage these elegantly and efficiently.

Acknowledgments and Disclosure of Funding

This work was supported by the German Federal Ministry of Education and Research (BMBF): Tübingen AI Center, FKZ: 01IS18039A. This work is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 409792180 (Emmy Noether Programme, project: Real Virtual Humans). GPM is a member of the Machine Learning Cluster of Excellence, EXC number 2064/1 – Project number 390727645. The project was made possible by funding from the Carl Zeiss Foundation.

References

[1] R. Alp Guler, G. Trigeorgis, E. Antonakos, P. Snape, S. Zafeiriou, and I. Kokkinos. DenseReg: Fully convolutional dense shape regression in-the-wild. In *CVPR*, 2017. 3

[2] M. Andriluka, U. Iqbal, E. Insafutdinov, L. Pishchulin, A. Milan, J. Gall, and B. Schiele. PoseTrack: A benchmark for human pose estimation and tracking. In *CVPR*, 2018. 7, 18

[3] F. Baradel, M. Armando, S. Galaaoui, R. Brégier, P. Weinzaepfel, G. Rogez, and T. Lucas. Multi-HMR: Multi-person whole-body human mesh recovery in a single shot. In *ECCV*, 2024. 8, 9

[4] E. G. Bazavan, A. Zanfir, M. Zanfir, W. T. Freeman, R. Sukthankar, and C. Sminchisescu. HSPACE: Synthetic parametric humans animated in complex environments. *arXiv:2112.12867*, 2021. 7, 18

[5] Y. Ben-Shabat, X. Yu, F. Saleh, D. Campbell, C. Rodriguez-Opazo, H. Li, and S. Gould. The IKEA ASM dataset: Understanding people assembling furniture through actions, objects and pose. In *WACV*, 2021. 7, 18

[6] B. L. Bhatnagar, X. Xie, I. Petrov, C. Sminchisescu, C. Theobalt, and G. Pons-Moll. BEHAVE: Dataset and method for tracking human object interactions. In *CVPR*, 2022. 6, 17

[7] S. Bian, J. Li, J. Tang, and C. Lu. ShapeBoost: Boosting human shape estimation with part-based parameterization and clothing-preserving augmentation. In *AAAI*, 2024. 7, 8

[8] M. J. Black, P. Patel, J. Tesch, and J. Yang. BEDLAM: A synthetic dataset of bodies exhibiting detailed lifelike animated motion. In *CVPR*, 2023. 6, 8, 17

[9] T. A. Bobach. *Natural neighbor interpolation-critical assessment and new contributions*. PhD thesis, Technische Universität Kaiserslautern, 2009. 6

[10] F. Bogo, J. Romero, G. Pons-Moll, and M. J. Black. Dynamic FAUST: Registering human bodies in motion. In *CVPR*, 2017. 6, 17

[11] Z. Cai, D. Ren, A. Zeng, Z. Lin, T. Yu, W. Wang, X. Fan, Y. Gao, Y. Yu, L. Pan, F. Hong, M. Zhang, C. C. Loy, L. Yang, and Z. Liu. HuMMan: Multi-modal 4D human dataset for versatile sensing and modeling. In *ECCV*, 2022. 6, 17

[12] Z. Cai, W. Yin, A. Zeng, C. Wei, Q. Sun, Y. Wang, H. E. Pang, H. Mei, M. Zhang, L. Zhang, C. C. Loy, L. Yang, and Z. Liu. SMPLer-X: Scaling up expressive human pose and shape estimation. In *NeurIPS: Datasets and Benchmarks*, 2023. 1, 2, 3, 8, 9

[13] C. Cao, T. Simon, J. K. Kim, G. Schwartz, M. Zollhoefer, S.-S. Saito, S. Lombardi, S.-E. Wei, D. Belko, S.-I. Yu, et al. Authentic volumetric avatars from a phone scan. In *TOG (Proc. SIGGRAPH)*, 2022. 2

[14] D. Casas and M. Comino-Trinidad. SMPLitex: A generative model and dataset for 3D human texture estimation from single image. In *BMVC*, 2023. 17

[15] P. Chandran, G. Zoss, P. Gotardo, and D. Bradley. Continuous landmark detection with 3D queries. In *CVPR*, 2023. 3

[16] W. Cheng, S. Xu, J. Piao, C. Qian, W. Wu, K.-Y. Lin, and H. Li. Generalizable neural performer: Learning robust radiance fields for human novel view synthesis. *arXiv:2204.11798*, 2022. 6, 17

[17] W. Cheng, R. Chen, W. Yin, S. Fan, K. Chen, H. He, H. Luo, Z. Cai, J. Wang, Y. Gao, Z. Yu, Z. Lin, D. Ren, L. Yang, Z. Liu, C. C. Loy, C. Qian, W. Wu, D. Lin, B. Dai, and K.-Y. Lin. DNA-Rendering: A diverse neural actor repository for high-fidelity human-centric rendering. In *ICCV*, 2023. 7, 18

[18] J. Chibane, A. Mir, and G. Pons-Moll. Neural unsigned distance fields for implicit function learning. In *NeurIPS*, 2020. 2

[19] J. Cho, K. Youwang, and T.-H. Oh. Cross-attention of disentangled modalities for 3D human mesh recovery with transformers. In *ECCV*, 2022. 3, 8

[20] H. Choi, G. Moon, and K. M. Lee. Pose2Mesh: Graph convolutional network for 3D human pose and mesh recovery from a 2D human pose. In *ECCV*, 2020. 3

[21] V. Choutas, L. Müller, C.-H. P. Huang, S. Tang, D. Tzionas, and M. J. Black. Accurate 3D body shape regression using metric and semantic attributes. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 8

[22] E. Corona, G. Pons-Moll, G. Alenyà, and F. Moreno-Noguer. Learned vertex descent: A new direction for 3D human model fitting. In *ECCV*, 2022. 3

[23] M. Fabbri, F. Lanzi, S. Calderara, A. Palazzi, R. Vezzani, and R. Cucchiara. Learning to detect and track visible and occluded body joints in a virtual world. In *ECCV*, 2018. 7, 18

[24] Z. Fan, O. Taheri, D. Tzionas, M. Kocabas, M. Kaufmann, M. J. Black, and O. Hilliges. ARCTIC: A dataset for dexterous bimanual hand-object manipulation. In *CVPR*, 2023. 6, 17

[25] H.-S. Fang, J. Li, H. Tang, C. Xu, H. Zhu, Y. Xiu, Y.-L. Li, and C. Lu. AlphaPose: Whole-body regional multi-person pose estimation and tracking in real-time. *TPAMI*, 2022. 7, 18

[26] M. Fieraru, M. Zanfir, E. Oneata, A.-I. Popa, V. Olaru, and C. Sminchisescu. Three-dimensional reconstruction of human interactions. In *CVPR*, 2020. 7, 18

[27] M. Fieraru, M. Zanfir, E. Oneata, A.-I. Popa, V. Olaru, and C. Sminchisescu. Learning complex 3D human self-contact. In *AAAI*, 2021. 7, 18

[28] M. Fieraru, M. Zanfir, S.-C. Pirlea, V. Olaru, and C. Sminchisescu. AIFit: Automatic 3D human-interpretable feedback models for fitness training. In *CVPR*, 2021. 7, 18

[29] S. Ghorbani, K. Mahdaviani, A. Thaler, K. Kording, D. J. Cook, G. Blohm, and N. F. Troje. MoVi: A large multi-purpose human motion and video dataset. *PLOS One*, 16(6):e0253157, 2021. 7, 18

[30] S. Goel, G. Pavlakos, J. Rajasegaran, A. Kanazawa*, and J. Malik*. Humans in 4D: Reconstructing and tracking humans with transformers. In *ICCV*, 2023. 1, 2, 8

[31] R. A. Güler, N. Neverova, and I. Kokkinos. DensePose: Dense human pose estimation in the wild. In *CVPR*, 2018. 3, 7, 18

[32] V. Guzov, I. A. Petrov, and G. Pons-Moll. Blendify – Python rendering framework for Blender. *arXiv:2410.17858*, 2024. 25

[33] D. Ha, A. M. Dai, and Q. V. Le. Hypernetworks. In *ICLR*, 2017. 2

[34] Y.-T. Hu, H.-S. Chen, K. Hui, J.-B. Huang, and A. G. Schwing. SAIL-VOS: Semantic amodal instance level video object segmentation – a synthetic dataset and baselines. In *CVPR*, 2019. 7, 18

[35] C.-H. P. Huang, H. Yi, M. Höschle, M. Safroshkin, T. Alexiadis, S. Polikovsky, D. Scharstein, and M. J. Black. Capturing and inferring dense full-body human-scene contact. In *CVPR*, 2022. 6, 17

[36] Y. Huang, O. Taheri, M. J. Black, and D. Tzionas. InterCap: Joint markerless 3D tracking of humans and objects in interaction. In *GCPR*, 2022. 6, 17

[37] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele. DeeperCut: A deeper, stronger, and faster multi-person pose estimation model. In *ECCV*, 2016. 7, 18

[38] S. Ioffe. Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. In *NIPS*, 2017. 19

[39] C. Ionescu, F. Li, and C. Sminchisescu. Latent structured models for human pose estimation. In *ICCV*, 2011. 18

[40] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *TPAMI*, 36(7):1325–1339, 2014. 6, 18

[41] S. Jin, L. Xu, J. Xu, C. Wang, W. Liu, C. Qian, W. Ouyang, and P. Luo. Whole-body human pose estimation in the wild. In *ECCV*, 2020. 7, 18

[42] G. Jocher, A. Chaurasia, and J. Qiu. Ultralytics YOLO, Jan. 2023. 18

[43] H. Joo, T. Simon, X. Li, H. Liu, L. Tan, L. Gui, S. Banerjee, T. Godisart, B. C. Nabbe, I. A. Matthews, T. Kanade, S. Nobuhara, and Y. Sheikh. Panoptic Studio: A massively multiview system for social interaction capture. *TPAMI*, 41(1):190–204, 2019. 6, 18

[44] W. Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 32:922–923, 1976. 2

[45] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik. End-to-end recovery of human shape and pose. In *CVPR*, 2018. 8

[46] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 2

[47] M. Kaufmann, J. Song, C. Guo, K. Shen, T. Jiang, C. Tang, J. J. Zárate, and O. Hilliges. EMDB: The Electromagnetic Database of Global 3D Human Pose and Shape in the Wild. In *International Conference on Computer Vision (ICCV)*, 2023. 6

[48] R. Khirodkar, A. Bansal, L. Ma, R. Newcombe, M. Vo, and K. Kitani. Egohumans: An egocentric 3D multi-human benchmark. In *ICCV*, 2023. 7, 18

[49] M. Kocabas, C.-H. P. Huang, O. Hilliges, and M. J. Black. PARE: Part attention regressor for 3D human body estimation. In *ICCV*, 2021. 8

[50] M. Kocabas, C.-H. P. Huang, J. Tesch, L. Müller, O. Hilliges, and M. J. Black. SPEC: Seeing people in the wild with an estimated camera. In *ICCV*, 2021. 6, 17

[51] 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 *ICCV*, 2019. 8, 18

[52] N. Kolotouros, G. Pavlakos, and K. Daniilidis. Convolutional mesh regression for single-image human shape reconstruction. In *CVPR*, 2019. 3

[53] 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 *CVPR*, 2017. 3

[54] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *JMLR*, 17 (39):1–40, 2016. 4

[55] J. Li, C. Xu, Z. Chen, S. Bian, L. Yang, and C. Lu. HybrIK: A hybrid analytical-neural inverse kinematics solution for 3D human pose and shape estimation. In *CVPR*, 2021. 8

[56] J. Li, S. Bian, C. Xu, Z. Chen, L. Yang, and C. Lu. Hybrik-x: Hybrid analytical-neural inverse kinematics for whole-body mesh recovery. *arXiv:2304.05690*, 2023. 8, 9

[57] R. Li, S. Yang, D. A. Ross, and A. Kanazawa. AI Choreographer: Music conditioned 3D dance generation with AIST++. In *ICCV*, 2021. 6, 18

[58] Y. Li, S. Si, G. Li, C.-J. Hsieh, and S. Bengio. Learnable Fourier features for multi-dimensional spatial positional encoding. *NeurIPS*, 2021. 5

[59] 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 *ECCV*, 2022. 8

[60] J. Lin, A. Zeng, S. Lu, Y. Cai, R. Zhang, H. Wang, and L. Zhang. Motion-X: A large-scale 3D expressive whole-body human motion dataset. In *NeurIPS*, 2023. 1

[61] J. Lin, A. Zeng, H. Wang, L. Zhang, and Y. Li. One-stage 3D whole-body mesh recovery with component aware transformer. In *CVPR*, 2023. 8, 9

[62] K. Lin, L. Wang, and Z. Liu. End-to-end human pose and mesh reconstruction with transformers. In *CVPR*, 2021. 3

[63] K. Lin, L. Wang, and Z. Liu. Mesh graphormer. In *ICCV*, 2021. 3

[64] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 18

[65] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. SMPL: A skinned multi-person linear model. *TOG (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, 2015. 2, 3

[66] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 6

[67] T. Luan, Y. Wang, J. Zhang, Z. Wang, Z. Zhou, and Y. Qiao. PC-HMR: Pose calibration for 3D human mesh recovery from 2D images/videos. In *AAAI*, 2021. 3

[68] X. Ma, J. Su, C. Wang, W. Zhu, and Y. Wang. 3D human mesh estimation from virtual markers. In *CVPR*, 2023. 3

[69] R. Martín-Martín, M. Patel, H. Rezatofighi, A. Shenoi, J. Gwak, E. Frankel, A. Sadeghian, and S. Savarese. JRDB: A dataset and benchmark of egocentric robot visual perception of humans in built environments. *TPAMI*, 2021. Early access. 18

[70] D. Mehta, H. Rhodin, D. Casas, P. Fua, O. Sotnychenko, W. Xu, and C. Theobalt. Monocular 3D human pose estimation in the wild using improved CNN supervision. In *3DV*, 2017. 6, 18

[71] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt. VNect: Real-time 3D human pose estimation with a single RGB camera. *TOG (Proc. SIGGRAPH)*, 36(4):44, 2017. 3

[72] D. Mehta, O. Sotnychenko, F. Mueller, W. Xu, S. Sridhar, G. Pons-Moll, and C. Theobalt. Single-shot multi-person 3D pose estimation from monocular RGB. In *3DV*, 2018. 6

[73] D. Mehta, O. Sotnychenko, F. Mueller, W. Xu, M. Elgharib, P. Fua, H.-P. Seidel, H. Rhodin, G. Pons-Moll, and C. Theobalt. XNect: Real-time multi-person 3D human pose estimation with a single RGB camera. *TOG (Proc. SIGGRAPH)*, 39(4), 2020. 3

[74] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *CVPR*, 2019. 2

[75] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2

[76] G. Moon and K. M. Lee. I2L-MeshNet: Image-to-lixel prediction network for accurate 3D human pose and mesh estimation from a single RGB image. In *ECCV*, 2020. 3

[77] G. Moon, H. Choi, and K. M. Lee. Accurate 3D hand pose estimation for whole-body 3D human mesh estimation. In *CVPR Workshops*, 2022. 8

[78] N. Neverova, R. A. Güler, and I. Kokkinos. Dense pose transfer. In *ECCV*, 2018. 3, 7, 18

[79] N. Neverova, D. Novotny, M. Szafraniec, V. Khalidov, P. Labatut, and A. Vedaldi. Continuous surface embeddings. *NeurIPS*, 2020. 3, 5

[80] A. Nibali, Z. He, S. Morgan, and L. Prendergast. Numerical coordinate regression with convolutional neural networks. *arXiv:1801.07372*, 2018. 4

[81] A. Nibali, J. Millward, Z. He, and S. Morgan. ASPset: An outdoor sports pose video dataset with 3D keypoint annotations. *Image and Vision Computing*, 111:104196, 2021. 6, 18

[82] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy. Berkeley MHAD: A comprehensive multimodal human action database. In *WACV*, 2013. 7, 18

[83] H. E. Pang, Z. Cai, L. Yang, T. Zhang, and Z. Liu. Benchmarking and analyzing 3D human pose and shape estimation beyond algorithms. In *NeurIPS: Datasets and Benchmarks*, 2022. 1

[84] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 2

[85] P. Patel, C.-H. P. Huang, J. Tesch, D. T. Hoffmann, S. Tripathi, and M. J. Black. AGORA: Avatars in geography optimized for regression analysis. In *CVPR*, 2021. 6, 17

[86] G. Pavlakos, X. Zhou, K. G. Derpanis, and K. Daniilidis. Coarse-to-fine volumetric prediction for single-image 3D human pose. In *CVPR*, 2017. 1, 3

[87] G. Pavlakos, V. Choutas, N. Ghorbani, T. Bolkart, A. A. A. Osman, D. Tzionas, and M. J. Black. Expressive body capture: 3D hands, face, and body from a single image. In *CVPR*, 2019. 6

[88] S. Peng, Y. Zhang, Y. Xu, Q. Wang, Q. Shuai, H. Bao, and X. Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *CVPR*, 2021. 6, 17

[89] A. Pumarola, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer. Unsupervised person image synthesis in arbitrary poses. In *CVPR*, 2018. 6, 18

[90] N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. Hamprecht, Y. Bengio, and A. Courville. On the spectral bias of neural networks. In *ICML*, 2019. 5

[91] R. M. Rustamov. Laplace–beltrami eigenfunctions for deformation invariant shape representation. In *Symposium on Geometry Processing*, 2007. 5

[92] I. Sárándi, T. Linder, K. O. Arras, and B. Leibe. MeTRAbs: Metric-scale truncation-robust heatmaps for absolute 3D human pose estimation. *IEEE Transactions on Biometrics, Behavior, and Identity Science (T-BIOM)*, 3(1):16–30, 2021. 1, 3, 4, 5

[93] I. Sárándi, A. Hermans, and B. Leibe. Learning 3D human pose estimation from dozens of datasets using a geometry-aware autoencoder to bridge between skeleton formats. In *WACV*, 2023. 1, 2, 3, 4, 6, 8, 19, 25

[94] R. Sarkar, A. Dave, G. Medioni, and B. Biggs. Shape of you: Precise 3D shape estimations for diverse body types. In *CVPR*, 2023. 8

[95] M. Seitzer, A. Tavakoli, D. Antic, and G. Martius. On the pitfalls of heteroscedastic uncertainty estimation with probabilistic neural networks. In *ICLR*, 2022. 20

[96] A. Sengupta, I. Budvytis, and R. Cipolla. Synthetic training for accurate 3D human pose and shape estimation in the wild. In *BMVC*, 2020. 3, 6, 8

[97] A. Sengupta, I. Budvytis, and R. Cipolla. Hierarchical kinematic probability distributions for 3D human shape and pose estimation from images in the wild. In *ICCV*, 2021. 8

[98] A. Sengupta, I. Budvytis, and R. Cipolla. Probabilistic 3D human shape and pose estimation from multiple unconstrained images in the wild. In *CVPR*, 2021. 7, 8

[99] A. Sengupta, I. Budvytis, and R. Cipolla. HuManiFlow: Ancestor-conditioned normalising flows on SO(3) manifolds for human pose and shape distribution estimation. In *CVPR*, 2023. 8

[100] R. Shapovalov, D. Novotny, B. Graham, P. Labatut, and A. Vedaldi. DensePose 3D: Lifting canonical surface maps of articulated objects to the third dimension. In *ICCV*, 2021. 3

[101] S. Shin, J. Kim, E. Halilaj, and M. J. Black. WHAM: Reconstructing world-grounded humans with accurate 3D motion. In *CVPR*, 2024. 8, 18

[102] R. Sibson. A brief description of natural neighbour interpolation. *Interpreting multivariate data*, pages 21–36, 1981. 6, 19

[103] J. Song, X. Chen, and O. Hilliges. Human body model fitting by learned gradient descent. In *ECCV*, 2020. 8

[104] X. Sun, B. Xiao, S. Liang, and Y. Wei. Integral human pose regression. In *ECCV*, 2018. 1, 3, 4

[105] Y. Sun, Q. Bao, W. Liu, Y. Fu, M. J. Black, and T. Mei. Monocular, one-stage, regression of multiple 3D people. In *ICCV*, 2021. 8

[106] M. Tan and Q. Le. EfficientNetV2: Smaller models and faster training. In *ICML*, 2021. 6

[107] M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 33:7537–7547, 2020. 5

[108] Y. Tian, H. Zhang, Y. Liu, and L. Wang. Recovering 3D human mesh from monocular images: A survey. *TPAMI*, 45(12):15406–15425, 2023. 2

[109] S. Tripathi, L. Müller, C.-H. P. Huang, T. Omid, M. J. Black, and D. Tzionas. 3D human pose estimation via intuitive physics. In *CVPR*, 2023. 6, 17

[110] M. Trumble, A. Gilbert, C. Malleson, A. Hilton, and J. Collomosse. Total Capture: 3D human pose estimation fusing video and inertial sensors. In *BMVC*, 2017. 7, 18

[111] N. P. van der Aa, X. Luo, G.-J. Giezeman, R. T. Tan, and R. C. Veltkamp. Utrecht Multi-Person Motion (UMPM) benchmark: a multi-person dataset with synchronized video and motion capture data for evaluation of articulated human motion and interaction. In *ICCV Workshops*, 2011. 7, 18

[112] G. Varol, J. Romero, X. Martin, N. Mahmood, M. J. Black, I. Laptev, and C. Schmid. Learning from synthetic humans. In *CVPR*, 2017. 6, 17

[113] E. Vendrow, D. T. Le, and H. Rezatofighi. JRDB-Pose: A large-scale dataset for multi-person pose estimation and tracking. In *CVPR*, 2023. 7, 18

[114] T. von Marcard, R. Henschel, M. J. Black, B. Rosenhahn, and G. Pons-Moll. Recovering accurate 3D human pose in the wild using IMUs and a moving camera. In *ECCV*, 2018. 6

[115] Z. Wang, L. Chen, S. Rathore, D. Shin, and C. Fowlkes. Geometric Pose Affordance: 3D human pose with scene constraints. *arXiv:1905.07718*, 2019. 6, 18

[116] J. Wu, H. Zheng, B. Zhao, Y. Li, B. Yan, R. Liang, W. Wang, S. Zhou, G. Lin, Y. Fu, et al. AI Challenger: A large-scale dataset for going deeper in image understanding. *arXiv:1711.06475*, 2017. 7, 18

[117] L. Xu, S. Jin, W. Liu, C. Qian, W. Ouyang, P. Luo, and X. Wang. ZoomNAS: Searching for whole-body human pose estimation in the wild. *TPAMI*, 2022. 18

[118] K. Yang, R. Gu, M. Wang, M. Toyoura, and G. Xu. LASOR: Learning accurate 3D human pose and shape via synthetic occlusion-aware data and neural mesh rendering. *IEEE Transactions on Image Processing*, 31:1938–1948, 2022. 8

[119] Z. Yang, Z. Cai, H. Mei, S. Liu, Z. Chen, W. Xiao, Y. Wei, Z. Qing, C. Wei, B. Dai, W. Wu, C. Qian, D. Lin, Z. Liu, and L. Yang. SynBody: Synthetic dataset with layered human models for 3D human perception and modeling. In *ICCV*, 2023. 6, 17

[120] P. Yao, Z. Fang, F. Wu, Y. Feng, and J. Li. DenseBody: Directly regressing dense 3D human pose and shape from a single color image. *arXiv preprint arXiv:1903.10153*, 2019. 3

[121] Y. Yin, C. Guo, M. Kaufmann, J. Zarate, J. Song, and O. Hilliges. Hi4D: 4D instance segmentation of close human interaction. In *CVPR*, 2023. 6, 17

[122] Y. Yoshiyasu. Deformable mesh transformer for 3D human mesh recovery. In *CVPR*, 2023. 3

[123] T. Yu, Z. Zheng, K. Guo, P. Liu, Q. Dai, and Y. Liu. Function4D: Real-time human volumetric capture from very sparse consumer RGBD sensors. In *CVPR*, 2021. 6, 17

[124] Z. Yu, J. S. Yoon, I. K. Lee, P. Venkatesh, J. Park, J. Yu, and H. S. Park. HUMBI: A large multiview dataset of human body expressions. In *CVPR*, 2020. 6, 18

[125] Y. Yuan, U. Iqbal, P. Molchanov, K. Kitani, and J. Kautz. GLAMR: Global occlusion-aware human mesh recovery with dynamic cameras. In *CVPR*, 2022. 8

[126] M. Zanfir, A. Zanfir, E. G. Bazavan, W. T. Freeman, R. Sukthankar, and C. Sminchisescu. THUNDR: Transformer-based 3D human reconstruction with markers. In *ICCV*, 2021. 3

[127] W. Zeng, W. Ouyang, P. Luo, W. Liu, and X. Wang. 3D human mesh regression with dense correspondence. In *CVPR*, 2020. 3

[128] H. Zhang, Y. Tian, X. Zhou, W. Ouyang, Y. Liu, L. Wang, and Z. Sun. PyMAF: 3D human pose and shape regression with pyramidal mesh alignment feedback loop. In *ICCV*, 2021. 8

[129] H. Zhang, Y. Tian, Y. Zhang, M. Li, L. An, Z. Sun, and Y. Liu. PyMAF-X: Towards well-aligned full-body model regression from monocular images. *TPAMI*, 2023. 8, 9

[130] S. Zhang, Q. Ma, Y. Zhang, Z. Qian, T. Kwon, M. Pollefeys, F. Bogo, and S. Tang. EgoBody: Human body shape and motion of interacting people from head-mounted devices. In *ECCV*, 2022. 6, 17

[131] T. Zhang, B. Huang, and Y. Wang. Object-occluded human shape and pose estimation from a single color image. In *CVPR*, 2020. 6, 18

[132] W. Zhang, Z. Liu, L. Zhou, H. Leung, and A. B. Chan. Martial Arts, Dancing and Sports Dataset: a challenging stereo and multi-view dataset for 3D human pose estimation. *Image and Vision Computing*, 61:22–39, 2017. 7, 18

[133] N. Zioulis and J. F. O'Brien. KBody: Towards general, robust, and aligned monocular whole-body estimation. In *CVPR*, 2023. 8

Appendix: Neural Localizer Fields for Continuous 3D Human Pose and Shape Estimation

In this appendix, we provide further technical details on

- the experimental setup (datasets, training configuration),
- uncertainty estimation evaluation,
- our proposed efficient parametric body model fitting algorithm (pseudocode, convergence analysis, fitting to subsets of vertices, uncertainty-based weighted fitting),
- our use of positional encodings,
- the neural field MLP architecture,
- qualitative results.

For video results, including visual comparison to prior work, we refer to our supplementary video. This video includes a demonstration of continuous querying in the canonical space.

A Mixture Dataset Description

The design goal of our work is to naturally mix data sources using different annotation formats. To show this capability, we perform mixed-batch training with batch size 160, where every batch contains examples with parametric annotations, examples with 3D keypoint annotations, examples with 2D keypoint annotations and examples with DensePose annotations. In the following, we describe the specific datasets we use for each of these types of annotations.

A.1 Parametric Annotations

We use the following datasets with parametric annotations, grouped by the body model:

- SMPL-X: AGORA [85], BEDLAM [8], RICH [35], MOYO [109], ARCTIC [24], InterCap [36], GeneBody [16], EgoBody [130]
- SMPL: SPEC [50], Hi4D [121], HuMMan [11], SynBody-NeRF [119], THuman2.0 [123], ZJU-Mocap [88], DFAUST [10], and SURREAL [112].
- SMPL+H: BEHAVE [6]

Rendering. DFAUST does not provide RGB images but contains diverse shapes that are useful for training, so we render 26K synthetic images with Blender, using randomized SMPLitex [14] textures, cameras and lights. Similarly, since Hi4D contains high-quality fits with good image-alignment, as well as high-quality textured meshes, we also render 100K synthetic images of these meshes with random camera angles and lights.

A.1.1 Shape Accuracy Considerations

Datasets have varying levels of shape accuracy, depending on the annotation pipeline used to create them, which means that we need to be careful about what we can treat as "ground truth" for training purposes.

Several datasets include annotations in SMPL(-X) format, but these were obtained by fitting *only to sparse skeletal body joint locations*, which were in turn triangulated from 2D skeletons (obtained e.g. from OpenPose). Therefore, the surface shape information in these fits does not line up well with the image, and the body shapes are biased towards the mean shape. To avoid similarly biasing our *model* to the mean shape, we only treat the joint annotations as valid in these datasets but do not use the surface. This applies to GeneBody, HuMMan and EgoBody.

Some datasets provide both the SMPL(-X) fit and raw body joints, either from triangulation or motion capture systems. We observed that the raw body joints tend to line up better with the image, so we ignore the SMPL(-X) fits in these cases. These datasets are: AIST-Dance++, DNA-Rendering, HUMBI and EgoHumans (these provide COCO-style keypoints besides the body fits) and BML-MoVi (which provides more than 80 markers on the body surface, as well as body joints – these exhibit better image-alignment than the provided SMPL fits).

A.2 3D Skeleton Annotations

We use skeleton annotations (non-SMPL-based) from the following dataset sources: Human3.6M [40, 39], MPI-INF-3DHP [70], MuCo-3DHP [70], HUMBI [124], 3DOH [131], CMU-Panoptic [43], AIST-Dance++ [57], ASPset510 [81], GPA [115], 3DPeople [89], SAIL-VOS [34], BML-MoVi [29], MADS [132], UMPM[111], Berkeley MHAD [82], TotalCapture [110], JTA [23], IKEA-ASM [5], Fit3D [28], CHI3D [26], HumanSC3D [27], HSPACE [4], EgoHumans [48], and DNA-Rendering [17].

Some of these datasets generated their annotations via multi-view triangulation of 2D pose estimation results, others used marker-based motion capture, and yet others are rendered through graphics and are annotated according to the skeleton of the underlying 3D assets, which may deviate from the SMPL skeleton.

A.3 2D Annotations

We are able to use training examples that only provide 2D keypoint information – in this case, the loss is computed solely by comparing the projection of our model's output to the ground-truth pixel positions. We use the following 2D keypoint datasets: MPII [37], COCO-WholeBody [64, 41, 117], JRDB-Pose [69, 113], PoseTrack [2], AI Challenger [116] and Halpe [25].

We also use the DensePose annotations from DensePose-COCO [31] and DensePose-PoseTrack [78].

B Evaluation Protocol

On 3DPW, we evaluate 14 joints, obtained through the same Human3.6M-style joint regressor that all prior works use. We only report evaluation results on the test set section of 3DPW.

Bounding boxes. For testing, some datasets provide bounding boxes (e.g., SSP-3D), while others do not (e.g., AGORA). In the latter case, we use the YOLOv8 [42] detector (trained on COCO). In the case of AGORA, we evaluate in the fine-tuned setting (NLF fine-tuned just on AGORA), and here we also fine-tune the YOLOv8 detector on AGORA-train.

Evaluation metrics. In our evaluations, we apply the conventionally used metrics for each individual benchmark. MPJPE is the average Euclidean joint error after alignment at the pelvis. (All distance-based errors are given as millimeters in this work.) Following prior works, e.g., [101], on 3DPW and EMDB the reference point is the midpoint between the two hip joints. P-MPJPE is the Euclidean error after Procrustes alignment, i.e., least-squares optimal rigid and uniform scaling alignment. When 14 joints are indicated for evaluation, these are obtained with a Human3.6M-like regressor from [51], which commonly used in the literature.

MVE is analogous to MPJPE but for mesh vertices instead of joints, similarly P-MVE is the Procrustes-aligned version of it.

MPJAE and P-MPJAE are the mean per joint angle errors without and with Procrustes alignment, in degrees. It is the geodesic distance on the $\mathrm{SO}(()3)$ rotation manifold, comparing the global body part orientations between ground truth and prediction.

On SSP-3D, PVE-T-SC evaluates the body shape, by computing the average Euclidean error for the predicted and the ground truth vertices in the default T-pose, after the prediction is optimally scale-aligned to the ground truth. The mIoU is the mean intersection over union metric comparing the binary masks obtained through projecting the ground truth mesh and the predicted mesh on the image plane.

On AGORA, NMVE is a normalized version of MVE that takes into account the detection performance as well. It is the product of MVE and the F1 measure (harmonic mean of precision and recall). Generally, MVE can be improved by not making predictions for hard cases, and the F1 penalty is aimed to compensate for this.

On MPI-INF-3DHP and MuPoTS-3D, PCK stands for the percentage of correct keypoints, namely those predicted joints that are within 150 mm of the ground truth.

C Training Details

Definition of the canonical volume. The canonical space is defined in reference to a specific T-like pose of the default SMPL mesh. All other keypoints (as e.g. shown in the referenced Fig. 4 column 1) are represented in this same coordinate system, as points within this canonical human volume. No explicit conversion between formats is necessary.

Canonical positions for 3D skeletal joints. The skeleton-based datasets do not provide their correspondence to our SMPL-compatible canonical space. Hence, we perform an approximate initialization and then treat the canonical position of each of several hundred joints (across the skeleton types) as trainable variables during the main training.

One could obtain this initialization in various ways. For this work, we first trained a model for predicting the separate skeleton formats (similar to the separate-heads baseline in [93]). We then ran inference with this predictor on the SURREAL dataset and trained linear regressors to interpolate from SURREAL GT vertices to the predicted keypoints and applied this regressor to the canonical template to obtain the approximate initialization.

The corresponding left and right joints are forced to be symmetrically placed in canonical space. To avoid divergence, we impose a constraint that the canonical positions cannot move away more than 70 mm from their initial positions.

Continuous deformation of body models. In SMPL(-X), skinning weights are only provided for the discrete set of mesh vertices, so by default forward kinematics can only be applied to these points. To be able to transform any continuous point (e.g., inside the body), we perform natural neighbor interpolation [102]. To avoid having to compute this interpolation on the fly for hundreds of points per training example, we precompute the interpolation weights for $\sim$1 million quasi-randomly sampled points (according to the Sobol sequence). Then during training, we apply forward kinematics to all SMPL vertices and produce the transformed internal point locations by computing a weighted average of the mesh vertices (according to the pre-computed natural-neighbor interpolation weights).

Point sampling. Using the above-mentioned interpolation method, we can choose to supervise any set of points in the canonical volume for training examples annotated with high-quality parametric models. We use a total of 1024 points for each such training example, of which 384 are inside the body (sampled from a large, precomputed set of more than a million points) and 640 are on the surface. The surface points are sampled uniformly by picking a random mesh triangle (with probability proportional to the triangle areas) and sampling a random point within the triangle (through barycentric coordinates).

Batch normalization. We found that in our mixture-dataset setting, the batch normalization layers in EfficientNetV2 can accumulate statistics during training that do not work well at inference time. This applies especially to longer trainings of the large backbone. We found it effective to use batch renormalization [38] instead, with default settings. Batch renorm mitigates the discrepancy between the training and inference behavior of batch norm. For numerical stability (i.e., avoiding crashes due to NaN values), we also found that constraining the convolutional kernel norms (to a maximum value of 20) helped.

Training schedule. We initialize with the 3D pose estimator-based weights from [93], and initially, for 3000 steps, we freeze the image backbone weights and only train the neural field. Subsequently, we decay the backbone learning rate from 3×10^{-5} to 2×10^{-6} with exponential decay with a larger drop after 90% of the training is done.

Augmentations. We use random rotation, scaling, translation, truncation, color distortion, synthetic occlusion, random erasing and JPEG compression for data augmentation during training.

D Inference Speed Optimization

NLF's inference-time overhead (for predicting weights through the field MLP) can be eliminated by precomputing the weights once for a chosen set of canonical points. (Typically one wants to localize the same points, i.e. same skeleton formats, for many images.) For reference, in case of NLF-S with

Table 9: **Ablation of uncertainty estimation.** We measure the average error of joints and vertices of SMPL on the 3DPW benchmark, along with Pearson's correlation (PCC) between the predicted pointwise uncertainty and the true error. Naively optimizing the log-likelihood harms the mean predictions, but the β_{NLL} loss [95] reduces the degradation and improves uncertainty quality. (NLF-S nonparam., on full 3DPW, 24 joints)

	MPJPE↓	MVE↓	PCC-Joints↑	PCC-Vertices↑
Euclidean loss (no uncertainty)	**59.2**	**71.3**	–	–
Negative log-likelihood (NLL) loss	61.2	73.6	0.50	0.34
β-NLL loss [95] ($\beta = 1$)	59.4	71.5	**0.54**	**0.40**

no image batching, and about 8000 points to be predicted (mesh vertices and skeletons), forwarding the field MLP to obtain convolution weights takes 7.7 ms, while the rest of the network including the backbone takes 12.7 ms. For NLF-L with batch size 64 the latter takes 587 ms, making the MLP cost negligible in comparison even if we do not precompute it.

E Uncertainty Estimation

To also consider our uncertainty prediction σ in the loss computation, we can use the negative log-likelihood loss as

$$\mathcal{L}_{\text{p,NLL}} = \frac{\|\hat{\mathbf{p}}' - \mathbf{p}'\|}{\sigma} + \log \sigma. \tag{10}$$

However, we noticed that this significantly diminishes the prediction quality for localization compared to the variant without predicting uncertainties. We attribute this to the effect described by Seitzer et al. [95], whereby the NLL loss downweights challenging training examples and impedes the learning progress. We therefore also experiment with applying their proposed β-NLL loss, which looks as follows when adapted to our context:

$$\mathcal{L}_{\text{p},\beta-\text{NLL}} = \left(\frac{\|\hat{\mathbf{p}}' - \mathbf{p}'\|}{\sigma} + \log \sigma \right) \cdot \text{sg}(\sigma)^{\beta}, \tag{11}$$

where $\text{sg}(\cdot)$ is the "stop gradient" operation that blocks the gradient flow during backpropagation. Intuitively, for $\beta = 1$, multiplication by $\text{sg}(\sigma)$ "cancels out" the division of $\|\hat{\mathbf{p}}' - \mathbf{p}'\|$ by σ, resulting in the same gradients for $\hat{\mathbf{p}}$ as in the case of vanilla Euclidean loss (removing the above-mentioned downweighting of challenging examples), without impacting the gradients for σ (due to $\text{sg}(\cdot)$).

To evaluate these different options, we measure Pearson's correlation coefficient (PCC) between the predicted pointwise uncertainty and the true error for all joints and vertices of the SMPL body model on the 3DPW benchmark test set, using all 24 joints and 6890 vertices. As seen in Tab. 9, the standard NLL loss significantly harms the mean prediction quality, but β_{NLL} largely fixes this problem and also yields uncertainties that are better correlated with the true error.

To illustrate that the obtained uncertainties are visually plausible, we show an example in Fig. 5, where the occluded hand has high predicted uncertainty, in line with intuition.

F Baseline Architecture Comparison without Localizer Field

To emphasize the key importance of using a localizer field in tying together the heatmap-producing convolutional weights for nearby points of the body, we perform the following ablation. We create a baseline architecture which uses no localizer field, and instead defines separate learnable parameters for each SMPL vertex and each joint of each skeleton format.

As seen in Fig. 6, when using the baseline model for inference, the different skeleton format predictions are visibly inconsistent with each other and with the SMPL mesh (see e.g. the H36M arm outside the SMPL body in the lower example), since the weights to localize each point have no enforced relation to each other. This also results in more scattered vertex predictions (see e.g. the hand region). NLF, by contrast, ensures that the different skeletons are localized consistently with each other, and the spatial smoothness of the prediction is improved.

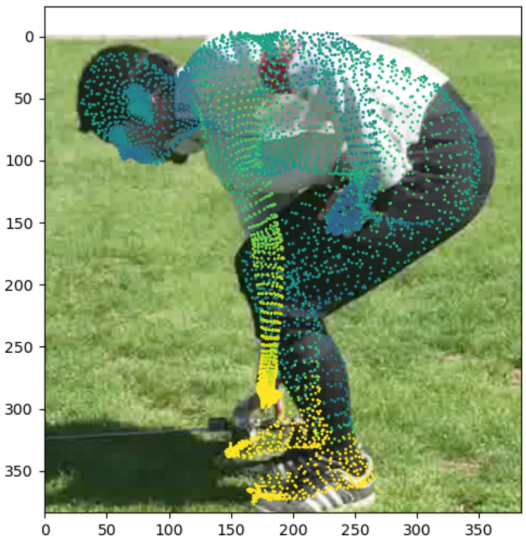

Figure 5: **Uncertainty estimation results.** High uncertainty is indicated in yellow, while low is shown in blue. Occluded body parts tend to have higher uncertainty prediction.

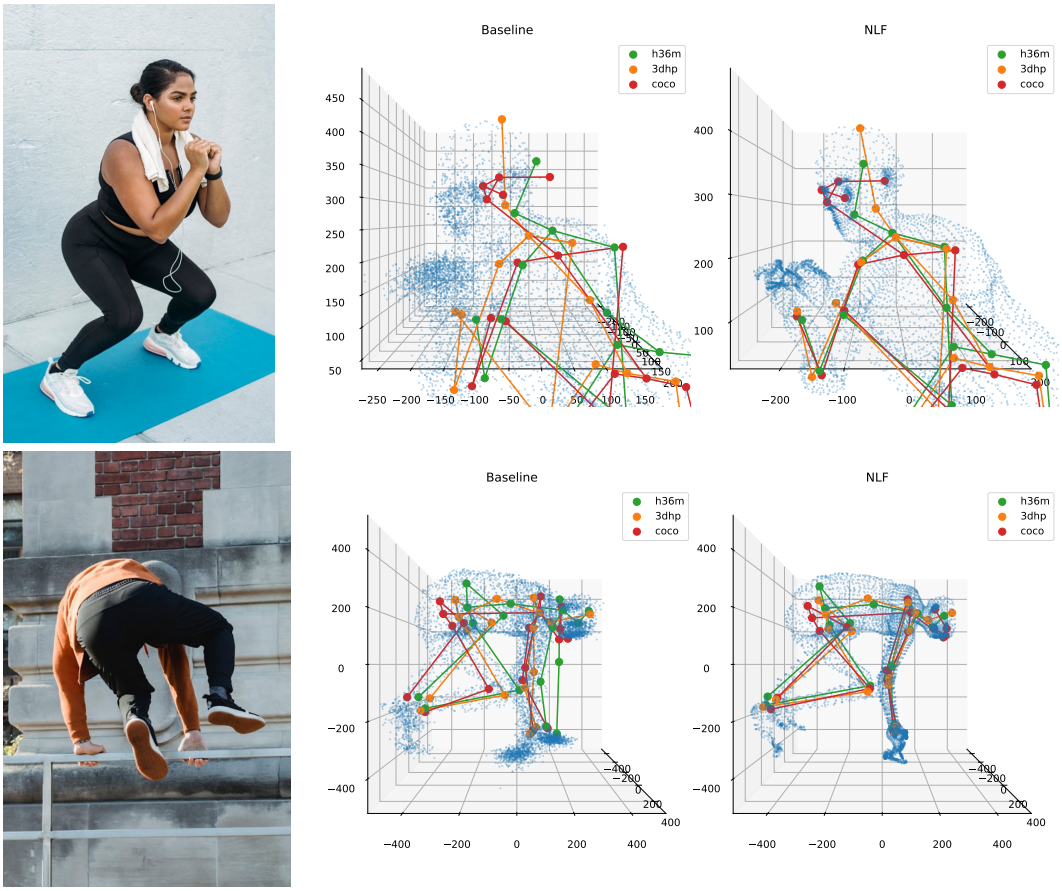

Figure 6: Side-views of the predictions by our NLF vs. a baseline architecture that trains separate, explicit convolutional weights for each skeletal joint and SMPL vertex.

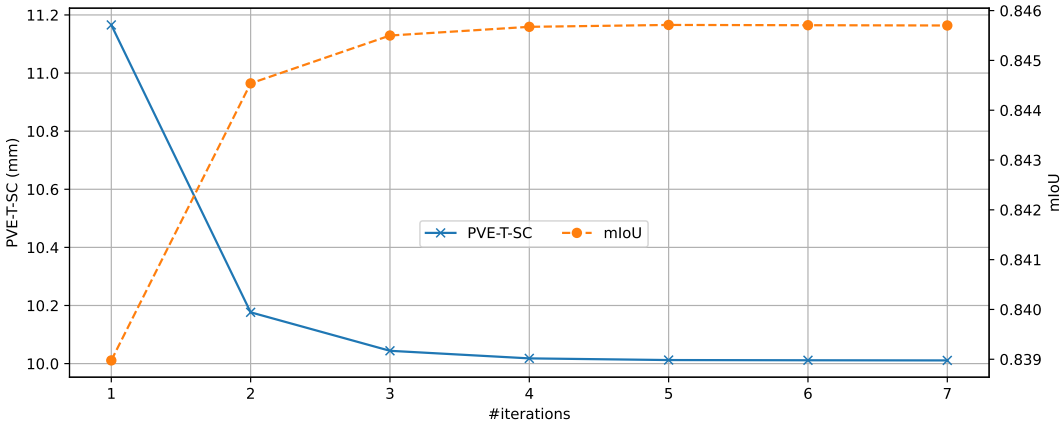

Figure 7: **Convergence of SMPL fitting.** We analyze the convergence properties of our iterative SMPL fitting algorithm on the SSP-3D benchmark. We use the nonparametric predictions from our NLF-S model as the fitting target. We can observe that approximately three iterations are sufficient for convergence to the SOTA result. (For PVE-T-SC lower is better, and for mIoU higher is better.)

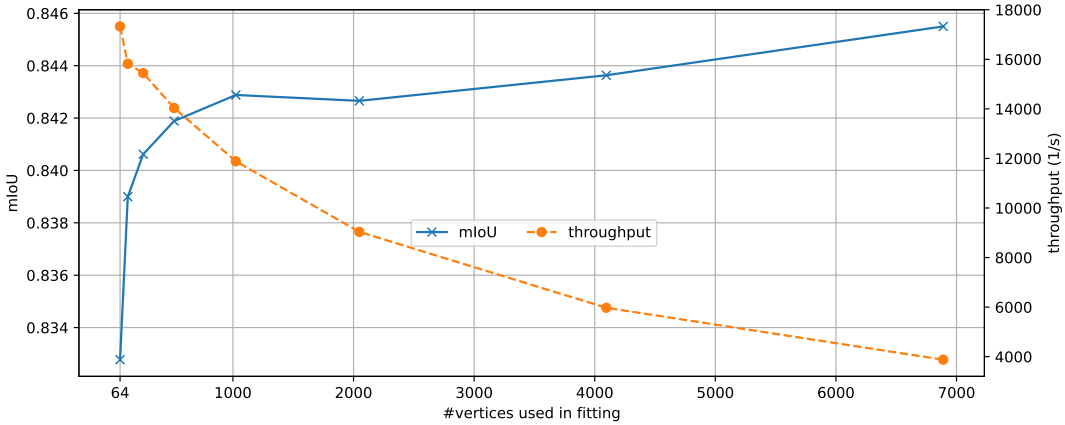

Figure 8: **Vertex-subset-based fitting of SMPL.** For additional efficiency, we evaluate fitting SMPL to only a subset of the 6890 vertices. The joints are still all used. To decide which vertices become part of each subset, we simplify the template mesh by quadric decimation and choose the vertices via distance-based Hungarian matching. We use 3 iterations in all cases.

G Efficient Body Model Fitting

In Algorithm 1, we provide the simplified pseudocode for our body model fitting algorithm used in the main paper. The following is an intuitive description of the steps. After initialization at the default T-pose, we iterate over the body parts. For each body part, we select the vertices and joints that form part of this body part. (Note that non-endeffector joints belong to two body parts as well.) We then use the (weighted) Kabsch algorithm to find the rotation matrix between the body part points in the current fit and in the input (fitting target).

Since most joints connect two body parts, it is important that our fit matches the orientation of the bone vectors as closely as possible, so that there is no error accumulation along the kinematic chain. We therefore weigh the joints much higher than the surface vertices in the rotation estimation. We specifically set a weight of 10^{-6} for vertices. Even though this weight is tiny, the vertices are meaningfully used as well, since many body parts consist of a single bone with only two joints, and fitting a rotation to two joints would leave one degree of freedom ambiguous (the rotation around the bone's main axis). By also including the vertices – albeit with a small weight – this additional degree of freedom is determined by the vertex positions. For body parts that contain at least three joints, the rotation fit is based practically solely on the position of these joints (e.g., the pelvis joint is the parent of three joints: two hips and one spine, making this body part contain four joints). Note that these estimated rotations are not parent-relative. Each body part's orientation is fitted independently of each other, to avoid any error accumulation.

Iterations	25	37	50	100
Time	3 min 18 s	6 min 54 s	16 min 35 s	33 min
Error	14.0 mm	8.0 mm	6.2 mm	5.0 mm

Table 10: Evaluation of the official SMPL-to-SMPLX converter according to total used time and achieved average error for 33 sample meshes under default settings and various maximum iteration counts. The time required is on the order of *minutes*, while our efficient body model fitter completes the same task with 7.8 mm error in about 28 *milliseconds*.

Table 11: **Ablation of uncertainty-based weights in body-model fitting.** Weighting points according to their estimated uncertainty brings modest benefits on 3DPW and EMDB (NLF-L model).

	3DPW-test (14 joints)				EMDB1 (24 joints)			
	MPJPE↓	P-MPJPE↓	MVE↓	P-MVE↓	MPJPE↓	P-MPJPE↓	MVE↓	P-MVE↓
w/o weighting	59.3	36.6	**69.6**	**48.8**	68.9	41.1	81.2	51.4
w/ weighting	**59.0**	**36.5**	69.7	**48.8**	**68.4**	**40.9**	**80.6**	**51.1**

The body model is then forwarded with the new orientation parameters, yielding new estimated fit vertices and joints, to be used in the second step.

In the second step, the β shape parameters and the translation vector $\mathbf{t}$ are estimated. Since SMPL uses linear blendshapes and linear blend skinning, the vertex and joint positions are a linear function of the shape parameters (while treating the pose rotation parameters as fixed). We simply need to compute the Jacobian matrix expressing this linear relationship (which involves traversing the kinematic tree). Given the Jacobian, we can solve for β (and $\mathbf{t}$) via L2-regularized linear least squares (solved via Cholesky decomposition). L2 regularization is used to avoid obtaining unrealistic shape vectors that have too large coefficients. We found that it is best not to penalize the first two components of β, which correspond to the person's general size (height and weight), in order to avoid biasing the process to an average-sized shape too strongly.

After obtaining the new shape estimate, we can re-estimate the per-body-part orientations, and this time the body parts have better shape correspondence, hopefully resulting in a more accurate orientation estimation. With the more accurate orientations, we can re-estimate the shape parameters, and so on.

As a final step, we re-estimate the orientations, but now not independently per body part (anchoring at the mean of each body part), but sequentially, traversing the kinematic chain, anchoring the rotation estimation of each body part at the position of its main joint (as determined by the previous rotation estimations). Here there is no reason to downweight the contribution of vertices, as the sequential estimation along the kinematic chain ensures that the body parts will link up correctly. This step reduces misalignment and error accumulation due to mismatched bone lengths between the shape fit and the nonparametrically estimated skeleton.

This process converges very quickly, as shown in Fig. 7, with the main improvement happening between the first and second iterations. From 3 iterations onward, convergence is reached.

To increase efficiency further, we may use only a subset of the vertices of the body model during fitting. In Fig. 8, we can see that, e.g., by fitting to only 1024 vertices, we increase the throughput by a factor of 3, while mIoU only decreases from 0.845 to 0.843. The timing was measured on an NVIDIA RTX 3090 GPU with a batch size of 2048.

In the **model transfer** experiment described in the main paper, where we fit SMPL-X parameters to clean SMPL mesh vertices with our fitting algorithm, we do not regularize the betas (as the input is a clean mesh without noise), use one iteration and 4096 vertices for fitting (out of the 10475). We also provide a more detailed comparison to the official model transfer code in Tab. 10, showing that our algorithm is much faster even at the same accuracy levels.

As described in the previous section, our algorithm outputs uncertainty estimates per point. These can be used to obtain a weighted version of the fitting algorithm. Specifically, given an estimated uncertainty σ_i for point i, we use the weighting factors $w_i \propto \sigma_i^{-1.5}$ when solving for the rotations and the shape coefficients. As shown in Tab. 11, this gives a slight improvement on both 3DPW and EMDB.

Algorithm 1 Efficient fitting of body model M to given target vertices V and joints J. $\Pi_{\mathrm{SO}(3)}$ denotes projection to $\mathrm{SO}(3)$ via SVD, the subscript $\{k\}$ selects joints or vertices that belong to body part k. $\mathcal{J}$ is the joint regressor. We use $\alpha = 10^{-6}$ in practice.

Require: $V \in \mathbb{R}^{N_v \times 3},\ J \in \mathbb{R}^{N_j \times 3},\ I \in \mathbb{N},\ \alpha \in [0, 1]$ ▷

$\tilde{V} \leftarrow V_0,\ \tilde{J} \leftarrow \mathcal{J}\tilde{V},\ \mathbf{t} \leftarrow \mathbf{0},\ R_k \leftarrow I_{3\times 3}$ ▷ Initialize the fit to mean-shaped T-pose

for $i \in [1..I]$ **do** ▷ 1–4 iterations

▷ (1) Keeping β and $\mathbf{t}$ fixed, solve for global rotations $\mathbf{R}$ via weighted Kabsch ◁

for $k \in [1..K]$ **do**

$P_{\{k\}} \leftarrow [V_{\{k\}},\ J_{\{k\}}]$ ▷ Concatenate vertices and joints together to a list of points

$\tilde{P}_{\{k\}} \leftarrow [\tilde{V}_{\{k\}},\ \tilde{J}_{\{k\}}]$

$W \leftarrow [\alpha,\ 1-\alpha]$ ▷ Weights for the vertices and joints, respectively

$\Sigma_k \leftarrow \mathrm{WeightedCov}\left(P_{\{k\}}, \tilde{P}_{\{k\}}, W\right)$ ▷ Weighted cov. mat. with vertices downweighted

$R_k \leftarrow \Pi_{\mathrm{SO}(3)}(\Sigma_k) R_k$ ▷ Project to the L2-nearest rotation matrix via SVD

$\tilde{V}, \tilde{J} \leftarrow M(\mathbf{R}, \beta) + \mathbf{t}$ ▷ Pose the vertices and joints with newly fitted $\mathbf{R}$

▷ (2) Keeping the rotations $\mathbf{R}$ fixed, solve for shape β and translation $\mathbf{t}$ ◁

$\tilde{P} \leftarrow [\tilde{V}, \tilde{J}]$ ▷ Concatenate vertices and joints together for uniform treatment

$P \leftarrow [V, J]$

Compute the Jacobian $\nabla_{\beta,\mathbf{t}} \tilde{P}$ via forward-mode autodiff. along the kinematic tree

$\beta, \mathbf{t} \leftarrow \left(\nabla_{\beta,\mathbf{t}} \tilde{P}\right) \setminus \left(P - \tilde{P}\right)$ ▷ Linear least squares (with regularization)

$\tilde{V}, \tilde{J} \leftarrow M(\mathbf{R}, \beta) + \mathbf{t}$ ▷ Update vertices and joints with new β, $\mathbf{t}$

▷ (3) Keeping β and $\mathbf{t}$ fixed, refine the rotations $\mathbf{R}$ by traversing the kinematic tree ◁

$\tilde{J}_1^{\text{new}} \leftarrow \tilde{J}_1$ ▷ Root position is not changed

for $k \in [1..K]$ **do** ▷ In topological order of the kinematic tree

$p \leftarrow \text{parent-index}(k)$

For non-root, determine $\tilde{J}_k^{\text{new}}$ from $\tilde{J}_p^{\text{new}}$, R_p and the T-pose k–p bone implied by β

$P_{\{k\}} \leftarrow [V_{\{k\}} - \tilde{J}_k^{\text{new}},\ J_{\{k\}} - \tilde{J}_k^{\text{new}}]$ ▷ Get the estimated body part relative to new pivot

$\tilde{P}_{\{k\}} \leftarrow [\tilde{V}_{\{k\}} - \tilde{J}_k,\ \tilde{J}_{\{k\}} - \tilde{J}_k]$ ▷ Get the fitted body part relative to its (old) pivot

$\Sigma_k \leftarrow {P_{\{k\}}}^T \tilde{P}_{\{k\}}$ ▷ "Covariance" anchored at the respective pivots instead of the means

$R_k \leftarrow \Pi_{\mathrm{SO}(3)}(\Sigma_k) R_k$

return $\mathbf{R}, \beta, \mathbf{t}$

We will make all code publicly available, including the fitting algorithm, to ensure its precise reproducibility.

H Neural Field Architecture and Details

Figure 9 shows the architecture of the neural field that parameterizes our point localizer network. The neural field has an MLP-like structure, starting with learnable Fourier features (fully connected layer followed by sine and cosine activations). After two further fully connected layers with a GELU activation inbetween, we arrive at the layer whose output is initially trained to approximate the global point signature (GPS) derived from the volumetric Laplacian. Further two FC layers with GELU inbetween yield the parameters to modulate the convolutional layer of the point localizer network as explained in the main paper.

We perform the following steps to obtain the global point signature and bake it into our neural field as initialization. We take the SMPL mesh in our canonical pose and finely subdivide it with Delaunay tetrahedralization. We process the resulting tetrahedral mesh with finite element methods to compute the eigenbasis of the Laplacian evaluated at each node of the tet mesh. These samples become the training set for training an MLP on top of learnable Fourier feature positional encoding. After this initial setup, the main training commences, where the weights of the neural field are tuned further, together with the backbone net.

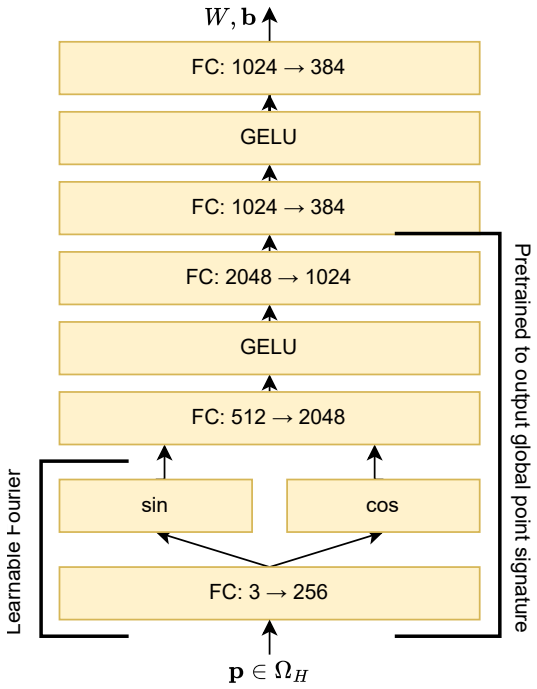

Figure 9: **Architecture of the neural localizer field itself.**

Table 12: **Results on 3DPW (24 joints).** In contrast to the main paper, here the whole 3DPW dataset used for evaluation, not only the part designated as test set, and all 24 joints are evaluated directly.

Method	MPJPE	P-MPJPE	MVE	P-MVE
MeTRAbs-ACAE-S [93]	60.6	41.7	–	–
NLF-S	**59.2**	**41.0**	71.7	52.0
NLF-S +fit	59.4	41.5	**71.1**	**50.8**
MeTRAbs-ACAE-L [93]	**58.9**	**39.5**	–	–
NLF-L	59.2	**39.5**	71.0	49.2
NLF-L +fit	59.0	40.0	**70.1**	**48.3**

I Additional 3DPW Protocol

For fair comparison to [93], in Tab. 12, we provide 3DPW evaluations matching their evaluation, i.e., using the entire dataset as the test set (and none of it for training), and evaluating all 24 joints directly (without applying a Human3.6M-like 14-joint regressor).

J Further Qualitative Results

In Figs. 10 and 11 we show further results of NLF-L with SMPL-fitting in post-processing (rendered using the Blendify framework [32]).

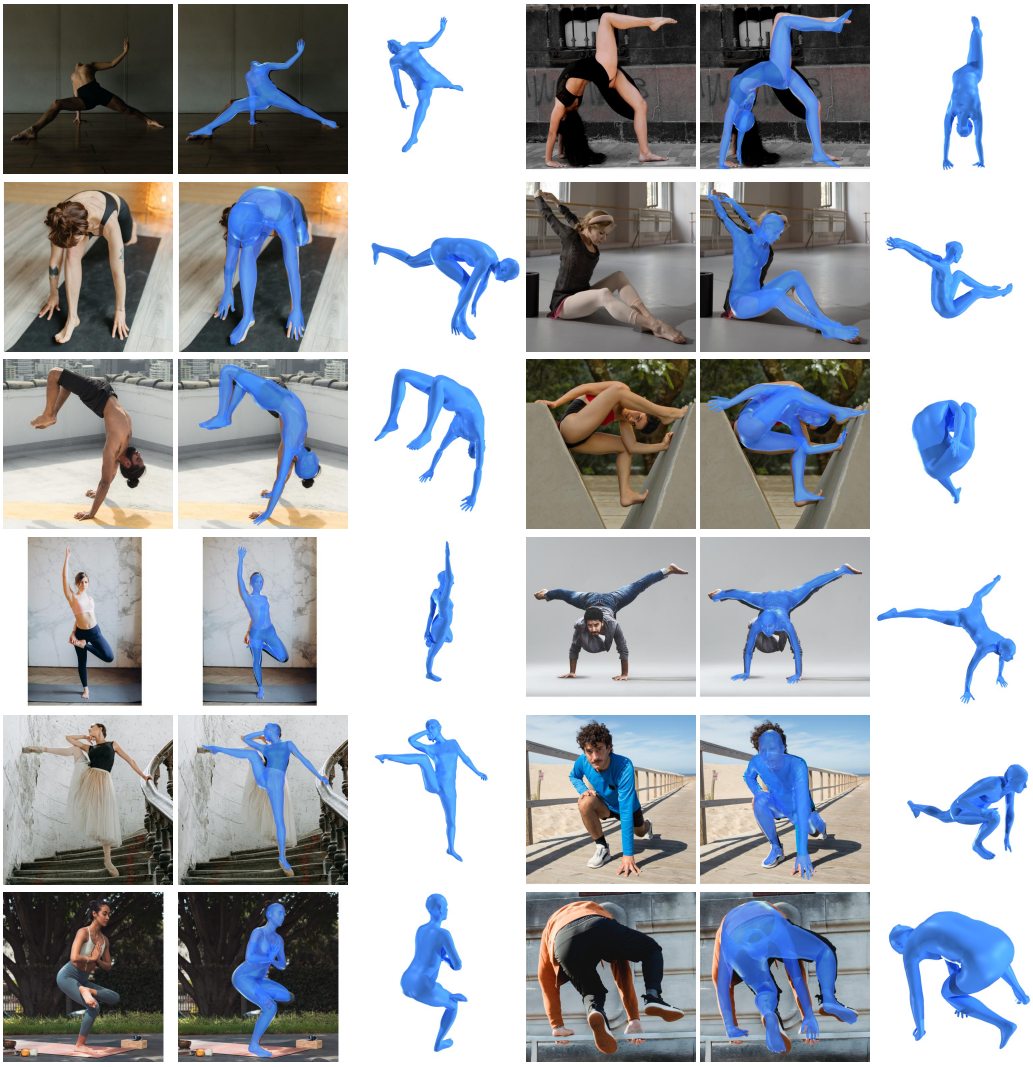

Figure 10: Qualitative results in the wild.

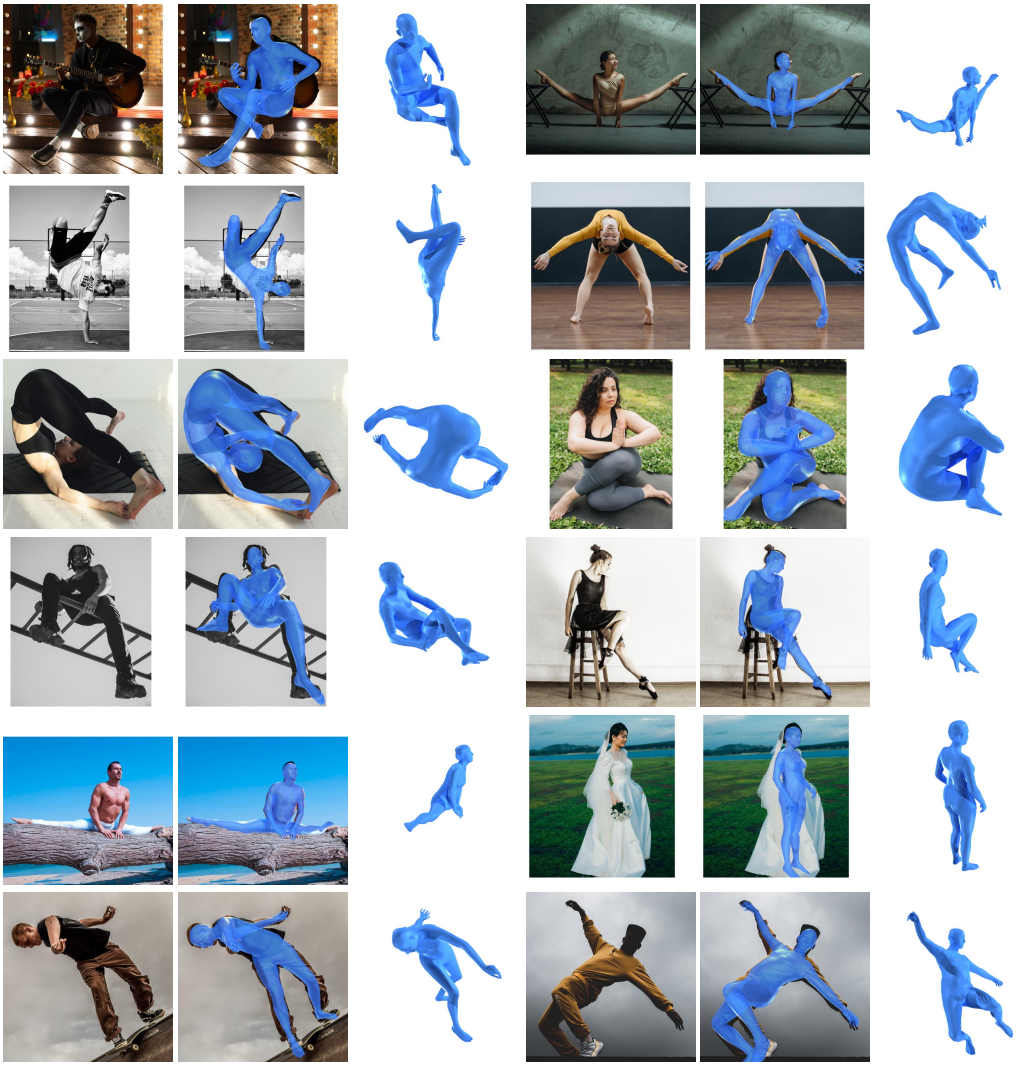

Figure 11: Qualitative results in the wild.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We substantiate each claim through experimental evaluation.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed at the end of the paper.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations"section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: This paper is focused on experimental work and does not prove theorems or other theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe experimental details in the appendix. Code release will ensure the details of reproducibility.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We will make our code and models available for research use before the conference takes place. We will also provide our data processing scripts. However, the code is not part of the submission package at this time.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide all necessary details for understanding the method and its impact, and will provide the full code to exactly specify the minor details.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We specify the standard error of the mean for MPJPE on 3DPW. It is also worth noting that we carefully seed the pseudorandom number generators the same way between trainings, and we verify that we obtain the same weight initialization, the same training example order and the same augmentation sequence in all runs. This reduces the experimental noise, leading to more reliable comparisons.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes"if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We specify the hardware (GPU type, memory and count) as well as the training time length in the main paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We have thoroughly reviewed and conform to the code of ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss positive and negative potential uses.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We do not release any dataset ourselves, and our model will only be made available for research use.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All used datasets are identified unambiguously through citations. We do not release any dataset ourselves.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not release new assets as part of the paper submission. Models will be made available exclusively for academic research purposes at a later time.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We perform no crowdsourcing nor research with human subjects. We use standard academic datasets, and did not collect our own data for this work.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.