NRDF: Neural Riemannian Distance Fields for Learning Articulated Pose Priors

Supplementary Material

In the following, we start with proving the propositions in Sec. A and discuss the relationship to Riemannian Flow Matching in Sec. A.3. Then, we present our user study about perceptual pose metrics in Sec. B, followed by implementation details in Sec. C and additional results in Sec. D.

A. Proofs & Theoretical Discussions

A.1. Proof of Prop. 1

Before proceeding with the proof let us recall the main proposition.

Proposition 1 (Quaternion-egrad2rgrad)**.** *For the quaternion manifold, the projection and mapping onto the tangent space of the canonical unit quaternion* $\mathbf{e} = \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}^\top$ *(egrad2rgrad in Eq. (2)) takes the form:*

$$\Pi_{\mathbf{q}}(\mathbf{v}) = \mathbf{P}\mathbf{v} - \frac{\mathbf{e}^\top \mathbf{P}\mathbf{v}}{1 + \mathbf{q}^\top \mathbf{e}}(\mathbf{q} + \mathbf{e}) \tag{15}$$

$$= \begin{bmatrix} 0 & 0 & 0 & 0 \\ -q_2/(1+q_1) & 1 & 0 & 0 \\ -q_3/(1+q_1) & 0 & 1 & 0 \\ -q_4/(1+q_1) & 0 & 0 & 1 \end{bmatrix} \mathbf{P}\mathbf{v}, \tag{16}$$

where $\mathbf{v} \in \mathbb{R}^4$ *and* $\mathbf{P} := \mathbf{P}(\mathbf{q}) = \mathbf{I} - \mathbf{q}\mathbf{q}^\top$.

Proof. First, note that the quaternion $\mathbf{q}$ is also the normal vector at point $\mathbf{q}$. Hence, the projection of any ambient vector onto the tangent space of a quaternion can be obtained by the standard projection onto the plane defined by the normal. In other words, the orthogonal complement of the tangent plane is the line in the direction of $\mathbf{q}$. Hence, we can project any ambient vector $\mathbf{v} \in \mathbb{H}_1$ onto $\mathcal{T}_{\mathbf{q}}\mathbb{H}_1$ as:

$$(\mathbf{v} - \mathbf{q}\mathbf{q}^\top \mathbf{v}) = (\mathbf{I} - \mathbf{q}\mathbf{q}^\top)\mathbf{v} = \mathbf{P}(\mathbf{q})\mathbf{v}. \tag{17}$$

Next, we need to rotate $\mathbf{P}(\mathbf{q})\mathbf{v}$ to align with the identity tangent space, $\mathcal{T}_{\mathbf{I}}\mathcal{M}$ or as in the proposition, $\mathcal{T}_{\mathbf{e}}\mathcal{M}$. To do so, we utilize the *discrete connection* on $\mathcal{S}^3$, which can be obtained by rotation of tangent planes. Since the tangent planes at $\mathbf{q}$ and $\mathbf{e}$ are defined by $\mathbf{q}$ and $\mathbf{e}$ themselves, all we need to do is to find the linear map, *i.e.* a rotation, that aligns $\mathbf{q}$ onto $\mathbf{e}$. This is the typical vector-rotation formula in 4-space and is given by:

$$\mathbf{v} - \frac{\mathbf{e}^\top \mathbf{v}}{1 + \mathbf{q}^\top \mathbf{e}}(\mathbf{q} + \mathbf{e}) \tag{18}$$

Plugging in $\mathbf{e} = \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}^\top$ and re-arranging yields the matrix form given in the proposition. Note that this is unique up to rotation around $\mathbf{e}$ and well connects to the non-uniqueness of parallel transport[1]. □

A.2. Proof of Prop. 2

We re-state Prop. 2 of the main paper before delving into the proof.

Proposition 2 (Distance preservation)**.** *Let* $\mathcal{P}$ *be a distribution over domain* $[0, 1]$, $\boldsymbol{\theta} \in \mathcal{D}$ *a data example,* $\hat{\boldsymbol{\theta}} \in \mathbb{H}_1^K$ *the output of Alg. 1 with input* $(\boldsymbol{\theta}, \mathcal{P})$, *and* $d = d(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}})$. *Then, for the distribution of resulting distances holds* $p(d) = \mathcal{P}$.

Proof. Our proof closely follows the steps in our algorithm where we sample a Gaussian vector of magnitude h, project it on the 3-sphere (where the distribution is uniform) and take steps. In the sequel, we prove each step.

Definition 9. *A vector* $\mathbf{v} \in \mathbb{R}^n$ *is said to be radially symmetric if* $\forall \mathbf{A} \in O(n), \mathbf{v} \overset{d}{=} \mathbf{A}\mathbf{v}$.

Lemma 1. *Let* $\mathbf{v} \in \mathbb{R}^n$ *be a* radially symmetric *random vector. Then* $\mathbf{v}/\|\mathbf{v}\| \sim \mathcal{U}(\mathcal{S}^{n-1})$.

Proof. Let $\mathbf{A} \in O(n)$ denote an orthogonal matrix and Proj be the projection onto $\mathcal{S}^{n-1}$. Then the following holds:

$$\mathrm{Proj}(\mathbf{v}) := \frac{\mathbf{v}}{\|\mathbf{v}\|} \overset{d}{=} \frac{\mathbf{A}\mathbf{v}}{\|\mathbf{v}\|} = \mathrm{Proj}(\mathbf{A}\mathbf{v}). \tag{19}$$

Thus, the random vector $\mathbf{v}/\|\mathbf{v}\|$ takes all its values over $\mathcal{S}^{n-1}$ and is radially symmetric. This is exactly the definition of $\mathcal{U}(\mathcal{S}^{n-1})$, the uniform distribution over $\mathcal{S}^{n-1}$. A more rigorous proof involves geometric measure theory, which is beyond the scope of this work. We refer the reader to [46] for further details. □

Showing that $\mathbf{v}$ after normalization follows a uniform distribution on the unit sphere in $\mathcal{T}_{\boldsymbol{\theta}}\mathcal{S}$, we move to the second part, where $\|\boldsymbol{\theta} - h \cdot \mathbf{v}\| = h$ since $\|\mathbf{v}\| = 1$ and $h \cdot \mathbf{v} \in \mathcal{T}_{\boldsymbol{\theta}}\mathcal{S}$ for any $h > 0$. In other words, h moves $\mathbf{v}$ towards/away from the base $\boldsymbol{\theta}$.

Lemma 2. *Let* $\mathbf{v} \in \mathcal{T}_{\boldsymbol{\theta}}\mathbb{H}_1^K$. *Using the ambient space of* $\mathbb{R}^4$ *and* $\|\mathbf{v}\| = 1$, *for all* t*:*

$$d\left(\boldsymbol{\theta}, \mathrm{Exp}_{\boldsymbol{\theta}}(h \cdot \mathbf{v})\right) = h\|\mathbf{v}\|, \tag{20}$$

in a sufficiently small interval (respecting the cut-locus), where $d(\cdot, \cdot)$ *denotes the geodesic distance.*

[1] The *hairy ball theorem* states the nonexistence of a global parameterization of any continuously varying basis vector of $\mathcal{T}_{\mathbf{x}}\mathcal{S}$ on all of $\mathcal{S}$.

Proof. The proof follows the do Carmo, "Riemannian Geometry", Proposition 3.6 [28]. □

Since by this lemma, the exponential map on the sphere preserves distances to the base within $[0, 1]$[2], it follows that $d(\boldsymbol{\theta}', \boldsymbol{\theta}) = h$. With a similar argumentation, when $h \sim \mathcal{P}$, $d(\boldsymbol{\theta}', \boldsymbol{\theta}) \sim \mathcal{P}$. □

How are the sample points distributed?. While our algorithm guarantees that the distances attained as a result of sampling preserve the initial choice of distance distribution, the distribution of the samples themselves can undergo distortion, *i.e.*, they can follow a complicated distribution on $\mathbb{H}_1^K$. Naturally, it is of interest to also understand how the samples are actually distributed on $\mathbb{H}_1^K$. While we cannot provide an explicit analytical form, we will provide an intuition into their distributions below, by following the exposition in [30].

Quaternions look like $SU(2)$, a Lie group isomorphic to $SO(3)$. Their Lie algebra (tangent space) $\mathfrak{su}(2)$ is isomorphic to the Lie algebra $\mathfrak{so}(3)$. Let us denote the Lie algebra of quaternions by $\mathfrak{h}$. Being the direct product of (non-Abelian) Lie groups, $\mathbb{H}_1^K$ is also a Lie group, with a Lie algebra $\mathfrak{h}^K$. The adjoint representation of $\mathbf{x} \in \mathfrak{h}^K$, $\mathfrak{ad}_{\mathbf{x}}(\mathbf{y})$, is the matrix representation of the map $[\mathbf{x}, \mathbf{y}]$ called the Lie bracket. For quaternions $\mathfrak{ad}_{\mathbf{x}}$ is the linear representation of this map. In our sampling algorithm (Alg. 1, to explicitly control the distance distribution, we first sample $\mathbf{v} \sim \mathcal{N}(0, \mathbf{I})$ in the Lie algebra of $\boldsymbol{\theta}$. We then normalize it yielding a uniform distribution on the 3-sphere and compose it with Gaussian sampled scalars h, resulting in a Gaussian distribution radially and uniform distribution in terms of direction: $\mathbf{v}/\|\mathbf{v}\| \sim \mathcal{U}(\mathcal{S}_h^3)$ and $\|\mathbf{v}\|_2 \sim \mathcal{N}(0, 1)$. We call this tangent distribution $r(\mathbf{v})$. Next, we use the exponential map to push r wrapping the samples on the manifold resulting in our *wrapped uniform-spherical distribution*:

$$p(\boldsymbol{\theta}) = \sum_{\substack{\mathbf{v} \in \mathfrak{h}^K \\ \mathrm{Exp}_{\boldsymbol{\mu}}(\mathbf{v}) = \boldsymbol{\theta}}} r(\mathbf{v}) \left| \mathbf{J}^{-1} \right|, \tag{21}$$

where $\mathbf{J}$ is the Jacobian map, whose determinant is the change of volume:

$$\left| \mathbf{J}^{-1} \right| := \det \left(\sum_{k=0}^{\infty} \frac{(-1)^k}{(k+1)!} \mathfrak{ad}_{\mathbf{x}}^k \right). \tag{22}$$

Intuitively, $\mathbf{J}$ linearly relates the tangent spaces and as such can be computed, similar to Eq. (9), via the parallel transport. As a result, we lose the simple form of the distribution in the tangent space and can only arrive at the final distribution via this push-forward operation.

[2]Note that, a common mistake is to assume that all distances are preserved. This is not true and only the distances to the base of the Exp/Log map is preserved.

A.3. Riemannian Flow Matching (RFM) vs. NRDF

Our training strategy resembles the extension of a recent state-of-the-art generative model, Flow Matching (FM) [44] onto Riemannian manifolds, known as, Riemannian Flow Matching (RFM) [21]. The differences lie in the sampling of training data and time steps as well as the way the *flow* is computed / predicted. In what follows, we will briefly summarize RFM and make the connection to our model NRDF. We will begin by recalling certain definitions.

Definition 10 (Riemannian CNF)**.** *A CNF $\varphi_t(\cdot) : \mathcal{M} \to \mathcal{M}$ on the smooth manifold $\mathcal{M}$ is defined by integration along a time-dependent vector field $v_t(\mathbf{x}) \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$: $\dot{\varphi}_t(\mathbf{x}) = v_t(\varphi_t(\mathbf{x}))$, parameterized by $t \in [0, 1]$, where $\varphi_0(\mathbf{x}) = (\mathbf{x})$.*

Definition 11 (Probability path)**.** *Let $\mathcal{P}(\mathcal{M})$ denote the space of probability distributions on $\mathcal{M}$. A probability path $\rho_t : [0, 1] \to \mathcal{P}(\mathcal{M})$ interpolates between two distributions $\rho_0, \rho_1 \in \mathcal{P}(\mathcal{M})$ indexed by $t \in [0, 1]$. ρ_t is said to be **generated** by φ_t if it* pushes forward *ρ_0 to ρ_1 following v_t,* i.e. *$\rho_t = [\varphi_t]_{\#}(\rho_0)$.*

Definition 12 (RFM)**.** *Given a probability path ρ_t, subject to the boundary conditions $\rho_0 = \rho_{\text{data}}$ and $\rho_1 = \rho_{\text{prior}}$, as well as an associated flow φ_t, learning a CNF by directly regressing v_t through a parametric, neural network g_β, is called Riemannian flow matching.*

Definition 13 (Riemannian Conditional FM)**.** *Unfortunately, the vanilla RFM objective is intractable as we do not have access to the closed-form u_t generating ρ_t. Instead, we can regress g_β against a tractable* conditional vector field *$v_t(\mathbf{x}_t \mid \mathbf{z})$, generating a* conditional probability path *$\rho_t(\mathbf{x}_t \mid \mathbf{z})$ which can recover the target unconditional path by marginalization:*

$$v_t(\mathbf{x}) = \int_{\mathcal{M}} v_t(\mathbf{x} \mid \mathbf{z}) \frac{\rho_t(\mathbf{x} \mid \mathbf{z}) q(\mathbf{z})}{\rho_t(\mathbf{x})} \, \mathrm{dvol}_{\mathbf{z}}. \tag{23}$$

Chen & Lipman then define the following Riemannian conditional FM (RCFM) objective for learning as:

$$\mathcal{L}_{RCFM}(\beta) = \mathbb{E}_{t, q(\mathbf{z}), \rho_t(\mathbf{x}_t \mid \mathbf{z})} d(g_\beta(t, \mathbf{x}_t), v_t(\mathbf{x}_t, \mathbf{z}))^2, \tag{24}$$

whose gradient is the same as that of RFM. Here, $t \in \mathcal{U}(0, 1)$ and $d(\cdot, \cdot)$ is the geodesic distance.

A simple variant of RCFM makes a particular choice of time scheduling, linearly decreasing the geodesic distance between $\mathbf{x}_t$ and $\mathbf{x}_1$ arriving at:

$$\mathbb{E}_{t, q(\mathbf{x}_1), p(\mathbf{x}_0)} \left\| v_t(\mathbf{x}_t, t) + d(\mathbf{x}_0, \mathbf{x}_1) \frac{\mathrm{grad}\, d(\mathbf{x}_t, \mathbf{x}_1)}{\|\mathrm{grad}\, d(\mathbf{x}_t, \mathbf{x}_1)\|_g^2} \right\|_g^2 \tag{25}$$

This form will closely relate to our work as we clarify below.

Algorithm 2 Training Riemannian Flow Matching for Learning on Articulated Bodies

Input: Base distribution $p(\boldsymbol{\theta}_0)$, target $q(\boldsymbol{\theta}_1)$, initial parameters ϕ_0 of a network g_ϕ
Output: Trained weights ϕ

while (not converged) **do**
  Sample $t \sim \mathcal{U}(0,1)$
  Sample training pose $\boldsymbol{\theta}_1 \sim q(\boldsymbol{\theta}_1)$
  Sample noisy pose $\boldsymbol{\theta}_0 \sim p(\boldsymbol{\theta}_0)$
  $\boldsymbol{\theta}_t = \mathrm{Exp}_{\boldsymbol{\theta}_0}(t \cdot \mathrm{Log}_{\boldsymbol{\theta}_0}(\boldsymbol{\theta}_1))$ (*cf*. Eq. (26))
  Update ϕ_t by minimizing $\mathcal{L}_{\mathrm{RCFM}}$ in Eq. (25)
end while

NRDF & RFM. In our work, we consider RFM on the product manifold of quaternions where $\mathcal{M} \equiv \mathbb{H}_1^K, \mathbf{x} \equiv \boldsymbol{\theta}$, and use the associated operators. We present in Alg. 2 the Riemannian Flow Matching adapted to our problem, articulated pose estimation. Note that, there are two fundamental differences: (i) the sampling of RFM and our sampling in Alg. 1, and (ii) the way the gradients are obtained. We now have a closer look into this.

Sampling. As seen in Alg. 2, RFM samples time uniformly, $t \sim \mathcal{U}(0,1)$, and a target pose is obtained as:

$$\boldsymbol{\theta}_t = \mathrm{Exp}_{\boldsymbol{\theta}_0}(t \cdot \mathrm{Log}_{\boldsymbol{\theta}_0}(\boldsymbol{\theta}_1)), \tag{26}$$

where $\boldsymbol{\theta}_1 \sim q(\boldsymbol{\theta}_1)$ is the data distribution and $\boldsymbol{\theta}_0 \sim p(\boldsymbol{\theta}_0)$ is the noise. Instead, our sampling algorithm presented in the main paper obtains a sample that is h away from a training pose as:

$$\hat{\boldsymbol{\theta}} \leftarrow \mathrm{Exp}_{\boldsymbol{\theta}}(h \cdot \mathbf{v}), \tag{27}$$

where $\mathbf{v}$ is directly sampled in the tangent space and normalized, and $h \sim \mathcal{P}$ for arbitrary $\mathcal{P}$ over $\mathbb{R}^+$. In contrast to RFM, we fix $d(\boldsymbol{\theta}, \mathbf{v}) = 1$ through normalization and thus, $d(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}) = h$ (explicit control over distance). Also note the time dependence. As our loss does not compute an explicit expectation over time, we can pre-compute all our training variables (nearest neighbors and the distances) offline. Having nearest neighbors as target distribution is unique to our work. As shown in Alg. 3, this greatly simplifies the training loop leading to stable and fast training. Moreover, note that, such pre-computation also allows for updating the nearest neighbor at each iteration during training.

Obtaining gradients. RFMs neural network minimizes Eq. (25) by explicitly predicting the steps. When we would use our sampler from above, $d(\boldsymbol{\theta}_0, \boldsymbol{\theta}_1) = 1$ would hold, and the network would be trained to the distance, whose derivation (by backpropagation) provides $\frac{\mathrm{grad}\ d(\boldsymbol{\theta}_t, \boldsymbol{\theta}_1)}{\|\mathrm{grad}\ d(\boldsymbol{\theta}_t, \boldsymbol{\theta}_1)\|_g^2}$, which is the gradient of the distance field. Thus, in this case, the gradient of our network would coincide with the prediction of the flow matching network. Note that even with the flow matching sampling, the flow matching network prediction points in the same direction as the distance field gradient, it is just scaled by $d(\boldsymbol{\theta}_0, \boldsymbol{\theta}_1) \neq 1$.

Algorithm 3 Neural Riemannian Distance Fields for Learning Articulated Pose Priors

Input: Distribution $\mathcal{P}$, target distribution $q(\boldsymbol{\theta})$, initial parameters ϕ of a network f_ϕ
Output: Trained weights ϕ

Get training data samples $\mathcal{D}'$ via Alg. 1
Search nearest neighbour $\boldsymbol{\theta}'$ and compute $d(\boldsymbol{\theta}, \boldsymbol{\theta}')$ for all $\boldsymbol{\theta} \in \mathcal{D}'$
while (not converged) **do**
  Sample a training pose $\boldsymbol{\theta}_i$ from $\mathcal{D}'$
  Update ϕ by minimizing
    $$\| f_\phi(\boldsymbol{\theta}_i) - \min_{\boldsymbol{\theta}' \in \mathcal{D}} d(\boldsymbol{\theta}_i, \boldsymbol{\theta}') \|$$
end while

B. User Study for Evaluation Metrics

Previous studies [19, 43] have highlighted a significant disparity between perceptual pose distance and commonly used metrics, such as differences in joint locations and orientations. The neural distance field model uses a certain distance metric to learn the relation between an arbitrary pose and the manifold of plausible poses by finding the nearest neighbor on the manifold. Consequently, NRDF relies on the distance metric possessing specific properties: 1) the distance metric is well-defined, and continuous and 2) the distance metric is close to human perception. To assess these criteria, we conducted a user study comparing various metrics, including orientation, Euclidean-based distance metrics, and a combination of both. We now define each distance metric used in our user study.

Orientation-based metrics. We take the distance metric used by Pose-NDF [63] as the candidate metric, denoting as $\Delta\mathbf{q}_p^l$. We also adopt he quaternion geodesic distance $\Delta\mathbf{q}_g^l$, which has a more explicit physical interpretation and covers a wider range of values. For both metrics, we further calculate the distance between noisy poses and their nearest neighbor in global (relative to the root) frames, denoted as $\Delta\mathbf{q}_p^g$ and $\Delta\mathbf{q}_g^g$, respectively.

Euclidean-based metrics. Our Euclidean-based metrics involve calculating the average Euclidean distance over all body joints and a specific set of surface markers, denoted as **j2j** and **m2m** respectively.

Combinations of orientation and Euclidean. In our observations, Euclidean-based metrics preserve the accurate overall shape of the body pose. However, they fall short of preserving the local twists of certain body joints. On the other hand, orientation-based metrics preserve precise local twists, yet they exhibit sensitivity to numerical values, resulting in divergent rotations even when the numerical values are close. To combine the strengths of both approaches, we introduce a hybrid metric, specifically defined as $\Delta\mathbf{q}+\mathrm{m2m} = \mathrm{m2m}+\lambda_\mathbf{q}\Delta\mathbf{q}_g^g$. This hybrid metric aims to leverage the advantages of Euclidean and orientation-based metrics, striking a balance that combines the faithful rep-

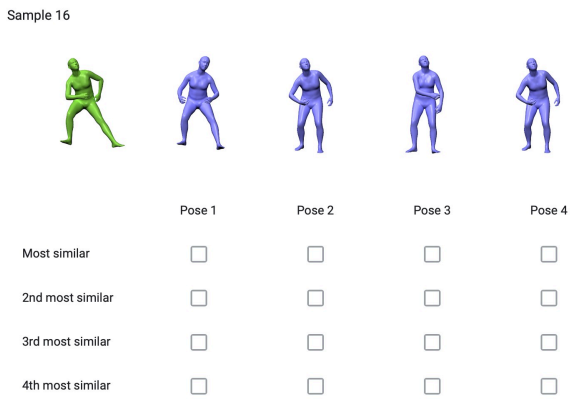

Figure 5. **User study for pose similarity assessment:** In our user interface, participants rank the similarity between a query pose (green) and its nearest neighbors (blue) from the AMASS dataset. These neighbors are obtained using different distance metrics.

resentation of the overall pose shape with the meticulous preservation of local joint twists. We set $\lambda_{\mathbf{q}} = 0.5$.

User study. We selected $\Delta\mathbf{q}_p^l$, $\Delta\mathbf{q}_g^g$, **m2m** and $\Delta\mathbf{q} + \text{m2m}$ as final candidates. We prepare 52 questions, each comprising the noisy pose and 4 NNs queried by a corresponding distance metric. Options are randomly shuffled in each question. As shown in Fig. 5, users ranked the options from most similar to least similar, with the flexibility to assign the same rank to multiple options.

Result analysis. From a total of 54 responses, 32.79% of users identified m2m as the most similar, while 30.09% favored $\Delta\mathbf{q} + \text{m2m}$. For the second most similar, 29.72% preferred $\Delta\mathbf{q} + \text{m2m}$, and 27.13% chose $\Delta\mathbf{q}_g^g$. Following this, we use m2m and $\Delta\mathbf{q} + \text{m2m}$ as evaluation metrics for the ablation studies.

C. Implementation Details

In this section, we introduce the experimental setup for data preparation, network training, baselines, and optimization-based downstream tasks such as partial-IK solver and image-based pose estimation.

C.1. Data Preparation

Training data. For training, we use a subset of the AMASS [45]. We follow the training split of AMASS used in VPoser [51] and Pose-NDF [63] and assume that the AMASS training set can sufficiently explain a comprehensive and valid human pose manifold.

To pre-process the AMASS dataset, we crop the central 80% of each motion sequence to focus on the most informative part of the data. We apply subsampling at a rate of 30 Hz, resulting in a total of 4 million clean poses. This is similar to VPosers and Pose-NDFs training setup. To create negative training samples along with their corresponding ground truth distances to the manifold, we sample noisy poses using Alg. 1, with $\mathcal{P} = \mathcal{N}(0, \pi/4)$. Following Pose-NDF [63], to speed up the NN search process, we adopt a multi-step mechanism for querying the NN of each noisy pose. For the first stage, we implement k'NN using FAISS [37], get k' candidates. For the second stage, we use the quaternion geodesic distance to find exact k neighbors from these k' candidates. In our implementation, we set $k' = 1000$ and $k = 1$. Note that we determine the final distance by considering only the closest NN, deviating from the approach in Pose-NDF [63], where the average distance over the top 5 NNs is computed. This is motivated by the observation that the top 5 NNs may exhibit discontinuities, and averaging their distances tends to over-smooth the manifold boundary.

Evaluation and validation. For validation, we utilize the validation split of the AMASS dataset, specifically we use Human Eva, MPI-HDM05, SFU, and MPI Mosh. For testing the accuracy and convergence speed across various baselines, we take the test split of AMASS dataset, specifically, we use SSM-Synced and Transitions. The distance values are computed in reference to the training data.

C.2. Network Training

Alg. 3 shows our training procedure. Specifically, we employ a hierarchical neural implicit network to implement NRDF, following the approach outlined in [63]. The network takes a quaternion representation of SMPL pose as input and produces a scalar distance field as output. We adopt a two-stage training strategy. For the first stage, each training batch comprises a balanced combination of 50% noisy poses and 50% clean poses. Subsequently, to enhance generalization to downstream tasks, in the second stage, we fine-tune our model using poses sampled from a standard normal distribution $\mathcal{N}(0, \mathbf{I}) \in \mathbb{R}^{4K}$. We set the learning rate to 1e-4. The entire training process requires 8-10 hours with a single GeForce RTX 3090 GPU.

C.3. Baseline Details

In this section, we present implementation details of baselines. Our evaluation focuses on the **pose denoising** task, where we compare finally converged poses with their ground truth nearest neighbors. We sample 20k noisy poses by using Alg. 1 based on AMASS [45] test set. We now first investigate the significance of our Riemannian projection (**Ours** v/s **Ours w/o RDFGrad**) and sampling method (**Pose-NDF** v/s **Ours w/o RDFGrad**). This is followed by comparison with a closely related Riemannian Flow Matching [21] based distance field (**Ours v/s FM-Dis**) and an ablation on distance v/s gradient field modeling.

Ours v/s Ours w/o RDFGrad. In this study, we evaluate the significance of our novel adaptive-step Riemannian gradient descent algorithm, termed as RDFGrad. For **Ours w/o RDFGrad** we use standard stochastic gradient descent (SGD) in Euclidean space. The results presented in Tab. 1

Figure 6. **Pose generation**: We compare pose generation results of our method with VPoser [51], GMM, FM-Dis, Pose-NDF [63], GAN-S [27], GFPose-A [25] and GFPose-Q. In comparison to VPoser, our method produces more diverse results. Furthermore, when compared to GMM, FM-Dis, and Pose-NDF, our method generates more realistic poses.

Method	Occ. Single Arm			Only End Effectors Visible			Occ. Legs		
	FID ↓	APD (in cm) ↑	d_{NN} (in rad) ↓	FID ↓	APD (in cm) ↑	d_{NN} (in rad) ↓	FID ↓	APD (in cm) ↑	d_{NN} (in rad) ↓
VPoser-Random	$1.145^{\pm.265}$	$3.085^{\pm.642}$	$0.066^{\pm.001}$	$0.681^{\pm.091}$	$8.330^{\pm.783}$	$0.067^{\pm.000}$	$0.748^{\pm.141}$	$6.650^{\pm.951}$	$0.058^{\pm.001}$
Pose-NDF [63]	$1.460^{\pm.233}$	$16.445^{\pm2.415}$	$0.622^{\pm.001}$	$2.081^{\pm..114}$	$31.524^{\pm.872}$	$0.738^{\pm.001}$	$3.015^{\pm.162}$	$24.831^{\pm1.328}$	$0.677^{\pm.001}$
FM-Dis	$1.150^{\pm.253}$	$4.967^{\pm.798}$	$0.210^{\pm.001}$	$1.010^{\pm.103}$	$10.812^{\pm1.451}$	$0.347^{\pm.002}$	$0.976^{\pm.158}$	$6.886^{\pm1.664}$	$0.226^{\pm.020}$
Ours	$1.306^{\pm.259}$	$5.892^{\pm.236}$	$0.132^{\pm.000}$	$0.964^{\pm.117}$	$10.388^{\pm.820}$	$0.137^{\pm.001}$	$0.899^{\pm.170}$	$6.705^{\pm.613}$	$0.125^{\pm.001}$

Table 5. **Quantitative results for IK Solver from partial/sparse joints.** We run all evaluations 20 times, $\pm$ indicates the 95% confidence interval. We evaluate under 3 settings: **Occ. Single Arm**, **Only End Effectors** (wrists and ankles) **Visible**, and **Occ. Legs**

illustrate that our approach achieves 6× acceleration in convergence speed thanks to the gradient update on the Riemannian quaternion manifold. Please refer to the supplementary video for qualitative comparisons. The convergence criterion is based on the absolute difference between the previously predicted distance and the current predicted distance being less than 1e-5.

Ours w/o (RDFGrad, d_q) v/s Pose-NDF. In this study we evaluate the significance of our novel training data sampling strategy. For this we compare **Pose-NDF**, which uses a naive sampling strategy with **Ours w/o (RDFGrad, d_q)**, which is basically Pose-NDF with our novel sampling strategy. From Tab. 1, we observe that just changing the sampling strategy, reduces the m2m distance from 25.04 cm to 15.16 cm, which shows the significance of the distance distribution of training data.

Ours v/s FM-Dis. In order to connect with the recent Riemannian flow matching, we introduce a new baseline, called **FM-Dis**, which extends RFM to model the pose prior as a distance field. Instead of predicting the gradient, our variation **FM-Dis** predicts the distance between $\boldsymbol{\theta}_t$ and the corresponding clean pose $\boldsymbol{\theta}_1$ without recomputing the nearest neighbor. Specifically, we minimize $\mathcal{L}_{\text{FM}-\text{Dis}}$, given by Eq. (29), where $\boldsymbol{\theta}_t$ is sampled by using Eq. (26),

$$\mathcal{L}_{\text{FM}-\text{Dis}}(\phi) \tag{28}$$

$$= \mathbb{E}_{t\sim\mathcal{U}(0,1),q(\boldsymbol{\theta}_1),p(\boldsymbol{\theta}_0)} \|v_\phi(\boldsymbol{\theta}_t) - d(\boldsymbol{\theta}_t, \boldsymbol{\theta}_1)\|_g^2 \tag{29}$$

It is apparent that $\boldsymbol{\theta}_t$ is evenly interpolated between the noise and a particular clean sample, which stands in contrast to our distribution, where the number of samples gradually decreases as one moves outward from the manifold. As shown in Fig. 6, FM-Dis tends to generate poses close to the mean. We show qualitative comparisons in our supplementary video.

Distance v/s gradient prediction. To connect our model with diffusion / score-based methods and flow matching-based methods, we implement **Gradient prediction w/o time**, **FM-Grad w/ time** and **GFPose-Q**. These approaches explicitly predict either the gradient or the gradient direction, while our method derives the gradient of distance with respect to the input pose through network back-propagation. Predicting full gradients (including length) without time-step conditioning is challenging for neural networks, leading to significant errors. Therefore, **Gradient prediction w/o time** is designed to predict the gradient direction (with normalized length) between the noisy pose and its nearest neighbor only. Noisy poses are sampled using Alg. 1 in this case. We additionally incorporate Riemannian flow matching (RFM) [21] into our experiment, denoting as **FM-Grad**. Different from RFM, we maximize the cosine similarity between the network prediction and gradient, thus, minimizing $\mathcal{L}_{\text{FM}-\text{Grad}}$, given by Eq. (30), where $t \sim \mathcal{U}(0,1)$ as above and $\boldsymbol{\theta}_t$ is obtained in the same manner as FM-Dis. We set $T = 1000$ during training.

$$\mathcal{L}_{\text{FM}-\text{Grad}}(\phi) \tag{30}$$

$$= -\mathbb{E}_{t,q(\boldsymbol{\theta}_1),p(\boldsymbol{\theta}_0)} \left| \frac{v_\phi(\boldsymbol{\theta}_t, t) \cdot \text{Log}_{\boldsymbol{\theta}_t}(\boldsymbol{\theta}_1)}{\|v_\phi(\boldsymbol{\theta}_t, t)\|_g^2 \|\text{Log}_{\boldsymbol{\theta}_t}(\boldsymbol{\theta}_1)\|_g^2} \right|.$$

For test-time projection, we follow Eq. (3) using $v_\phi(\boldsymbol{\theta}_t, t)$ as the gradient. We set $\tau = 0.01$ and the initial time-step $T' = 200$. For training Gradient prediction w/o time and FM-Grad, we employ the same network architecture as GFPose [25]. The convergence criterion is based on the absolute difference between the predicted gradient norm at $t+1$ and t being less than 1e-5. Regarding **GFPose-Q**, we retrain it using the quaternion representation on the AMASS dataset, with $\emptyset$ conditioning. Since gradient prediction is less accurate than distance prediction, results based on gradient prediction tend to exhibit either over-correction or unrealistic poses. Please refer to the supplementary video for qualitative comparisons.

C.4. IK Solver Setup

We utilize the AMASS test set and compute ground truth marker or joint positions through forward kinematics. The overall loss function, based on Eq. (12), encompasses a data term defined by the L2 loss between predicted marker/joint locations and observations. Given that most off-the-shelf optimizers in PyTorch are SGD-based or its variations, and there is no optimizer designed for quaternions in geoopt [41], we introduce a custom optimizer specifically designed for RDFGrad. This involves obtaining the Euclidean gradient returned by the network and projecting it onto the Riemannian quaternion manifold using Eq. 9. We plan to release our code for public use and stimulating future research. During optimization, for VPoser, VPoser-Random and Pose-NDF, we set $\lambda_{\boldsymbol{\theta}} = 0.1$, $\lambda_\beta = 0.05$ and $\lambda_\alpha = 0.001$. For FM-Dis and NRDF, we set $\lambda_{\boldsymbol{\theta}} = 5.0$, $\lambda_\beta = 0.05$ and $\lambda_\alpha = 0$. Concerning our RDFGrad-based

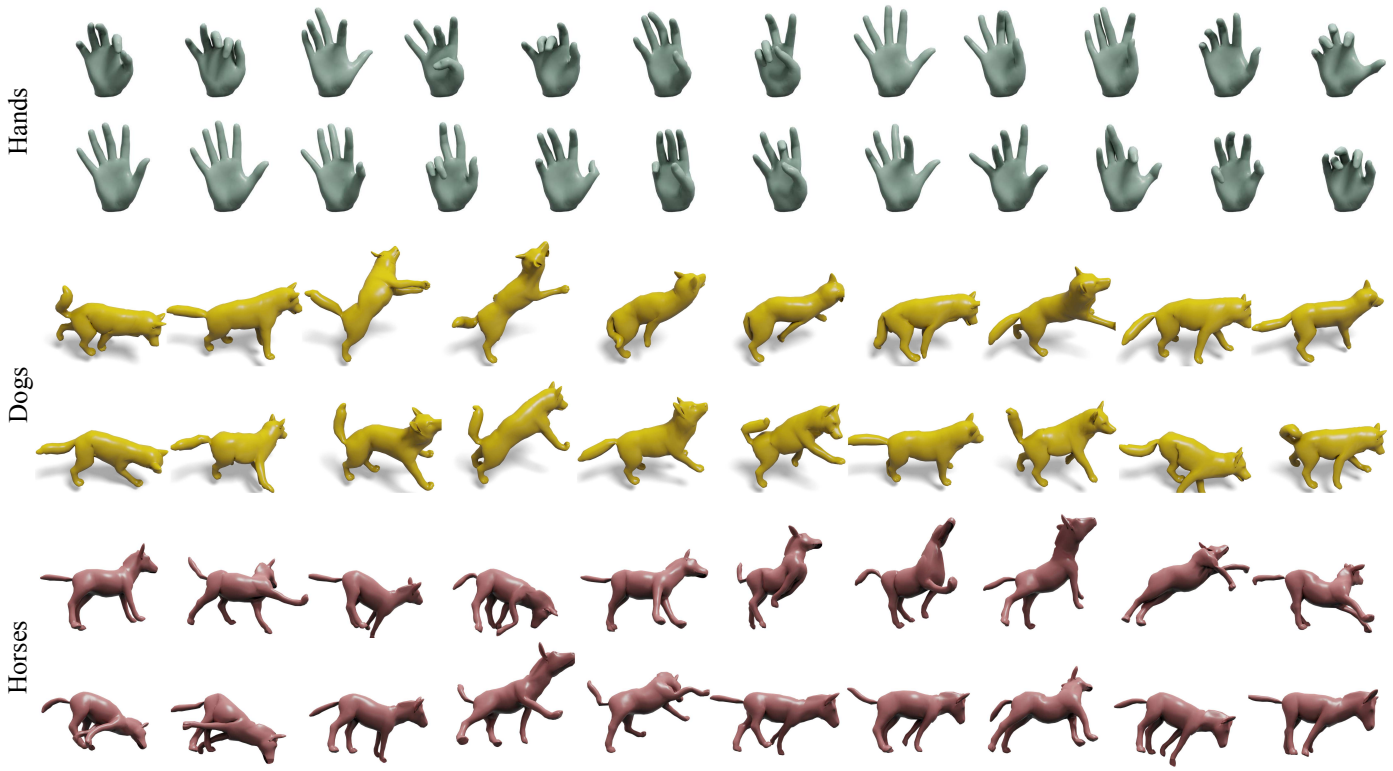

Figure 7. **Animal and Hand pose generation:** We generate diverse animal and hand poses using NRDF.

optimizer, for an effective initialization of the prior loss, we exclude the data term in the first stage and optimize only the prior term. Subsequently, we combine all loss terms for joint optimization. The stopping criterion for all experiments is set as MPJPE $= 3$ cm.

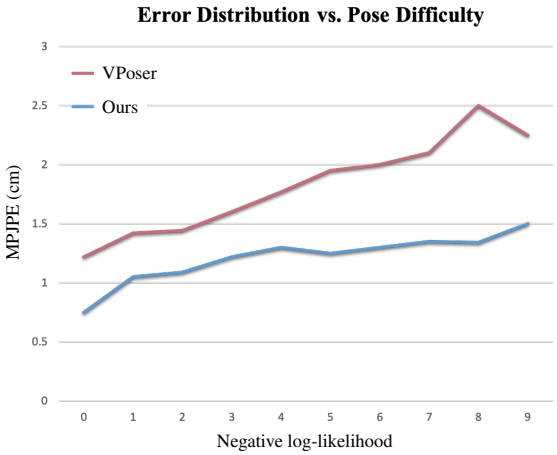

Figure 8. **Error Distribution vs. Pose Difficulty:** X axis represents the relative negative log likelihood (NLL) while Y axis represents the MPJPE between the result joint locations and corresponding observations.

C.5. Image-based Tasks Setup

For evaluating the impact of our prior on human pose estimation from images, we use 3DPW dataset [67]. We conduct the evaluation on the test split of 3DPW, where ground truth for a single person in the image is available. We crop the images using the GT 2d keypoints and discard the images where SMPLerX doesn't predict any person. This amounts to roughly 5k images. We use "SMPLer-X-S32" for predicting SMPLX parameters and use optimization-based processing to convert SMPLX parameters to SMPL [51]. In particular, we want to refine network prediction using optimization-based refinement. We use SMPLer-X [18] for predicting human pose $\hat{\boldsymbol{\theta}}$ and shape $\hat{\boldsymbol{\beta}}$ from images and then use optimization loss mentioned in Eq. (12) and Eq. (14) to refine the predictions. For VPoser, we optimize the latent vector z of VAE, where z is initialized from the VAE encoding of predicted pose or $\hat{\boldsymbol{\theta}}$. More specifically $z_{\text{init}} = f_{\text{VE}}(\hat{\boldsymbol{\theta}})$, where f_{VE} is encoder of VPoser. For Pose-NDF, FM-Dis, and Our prior optimization, we simply optimize for $\boldsymbol{\theta}$ and the variable is initialized using $\hat{\boldsymbol{\theta}}$. We also optimize for SMPL shape ($\boldsymbol{\beta}$) parameters in both setups. For the evaluation metric, we have used PA-MPJPE (Procrustes aligned-MPJPE), PA-PVE (Procrustes aligned-per-vertex error).

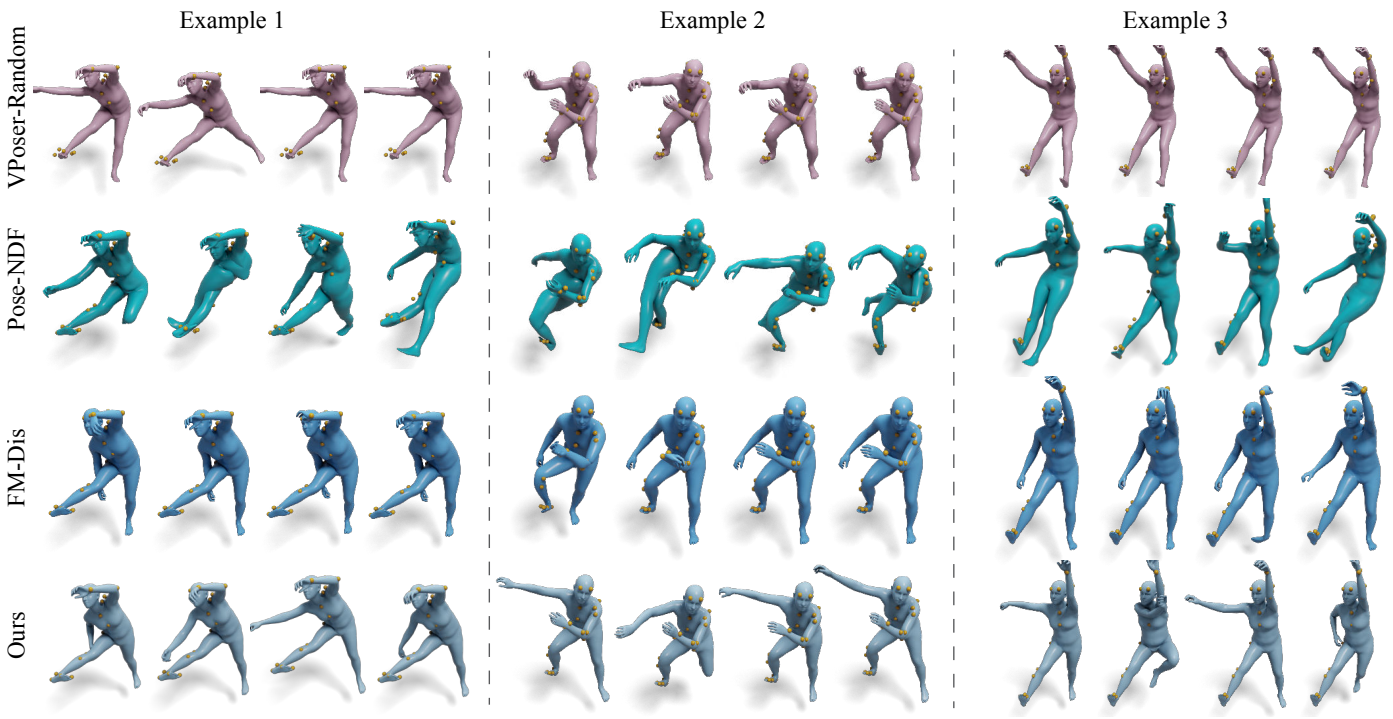

Figure 9. **IK Solver for one arm and one leg occluded:** Given partial observation, where one leg and one arm are occluded, we perform 3D pose completion. We observe that VPoser [51] based optimization generates realistic, yet very limited diversity in poses. Pose-NDF [63] generates more diverse, but unrealistic poses. FM-Dis also generates limited diversity in poses. NRDF generates diverse and realistic poses as compared to the aforementioned pose priors.

D. Additional Results

We now provide additional qualitative and quantitative results.

D.1. IK solver from partial surface markers or body joints

We show more IK results from partial observations. For surface markers, we observe that for Occ. Single Arm and Visible end-effectors setup, our model generates more diverse poses based on APD. We also show qualitative results in Fig. 10. Note that despite the numerical diversity of VPoser, it exhibits fewer diverse poses for occluded legs compared to NRDF. As depicted in Fig. 11, the legs of VPoser tend to move together without interaction between them, which could also result in a large APD value. In contrast, our method demonstrates more diverse leg poses, including bending of knees. We provide results for another setup in Fig. 9, where one arm and one leg are occluded. Given that body joints are more underconstrained and challenging, we further evaluate our IK solver on partial body joint observations. Tab. 5 and Fig. 12 illustrate the IK results, showcasing that our method achieves accurate IK while maintaining more diversity.

D.2. Monocular 3D Pose Estimation from Images

We provide more qualitative results for 3D pose estimation from images in Fig. 13.

D.3. More Pose Generation Results

Fig. 6 shows additional generation results from different methods. Note that poses generated by GMM can appear unrealistic and can have implausible bends around joints such as elbow or shoulder joints, as shown in Fig. 6 (top-right). VPoser [51] tends to generate more standing and less diverse poses. This is attributed to Gaussian assumption of the latent space. FM-Dis also generates less diverse poses, close to mean poses and some times results in unrealistic poses as well. Pose-NDF [63] generates diverse poses, in terms of bends around knees, elbows *etc*. but at the same time, it results in implausible poses. This is attributed to the inaccurate distance field of Pose-NDF. GAN-S [27] also tends to generate less diverse pose, as compared to Pose-NDF. We also compare with a diffusion-based model GFPose [25]. We retrain GFPose on AMASS dataset and call it GFPose-A. Since this is joint-location based model, we observe that the generated results might have inconsistent bone lengths, as highlighted in Fig. 6. We also retrain GFPose on quaternions, denoting as GFPose-Q, which similarly generates less diverse and unrealistic poses.

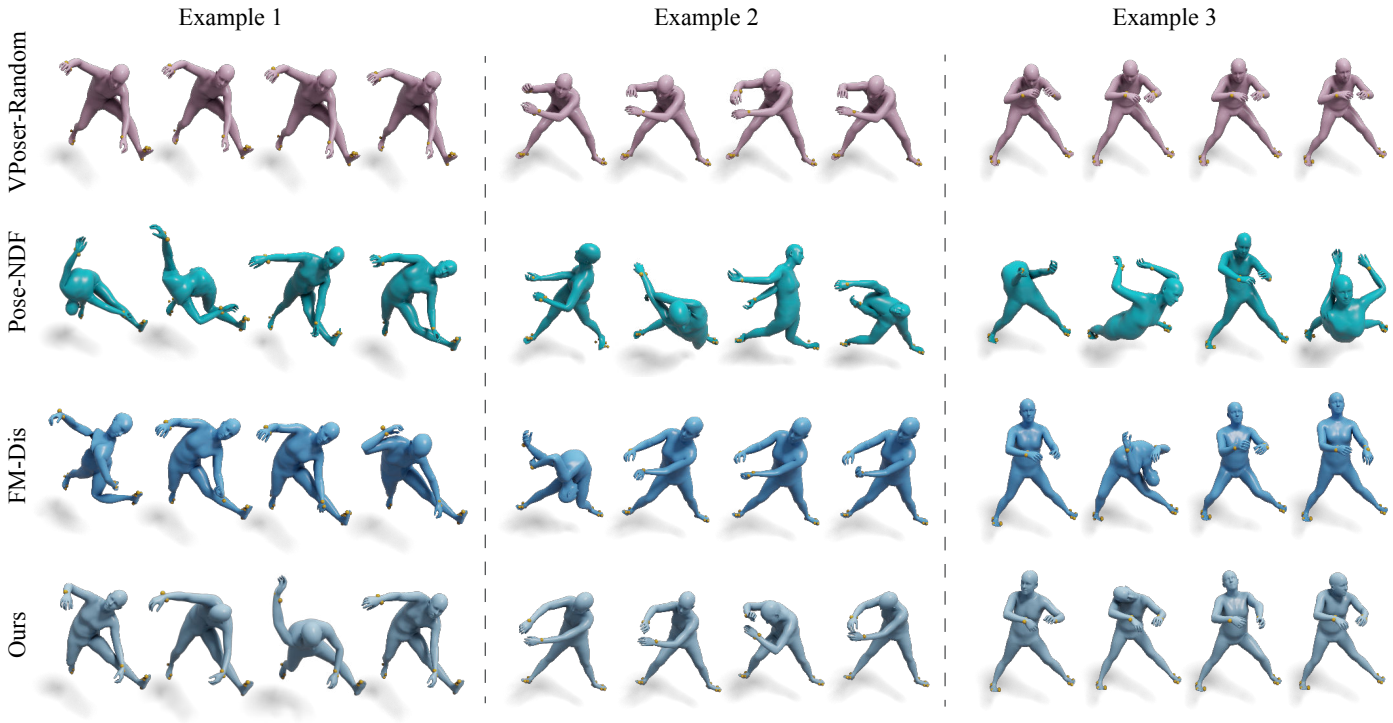

Figure 10. **IK Solver for visible end-effectors:** Given partial observation, where only end-effectors are visible (yellow markers), we perform 3D pose completion. We observe that VPoser [51] based optimization generates realistic, yet very limited diversity in poses or almost similar poses. Pose-NDF [63] and FM-Dis result unrealistic poses for such sparse observations. NRDF generates diverse and realistic poses as compared to the aforementioned pose priors.

We further show more results for hand and animal pose generation in Fig. 7.

D.4. Error Distribution vs. Pose Difficulty

Poses generated by VPoser [51] exhibit a tendency to cluster around mean poses, given it is based on Gaussian assumptions. In this subsection, we explore the correlation between location error and pose difficulty in partial Inverse Kinematics (IK) tasks. The observed relationship is visually depicted in Fig. 8. It's noteworthy that as the ground truth pose becomes less common (indicated by larger Negative Log-Likelihood (NLL) values), the difference between Mean Per Joint Position Error (MPJPE) of VPoser and NRDF tends to increase. However, NRDF consistently maintains a smaller error, remaining under 1.5 cm. The first column of Fig. 9 (foot part of VPoser) also shows that VPoser fails to meet the observations when the given pose is uncommon.

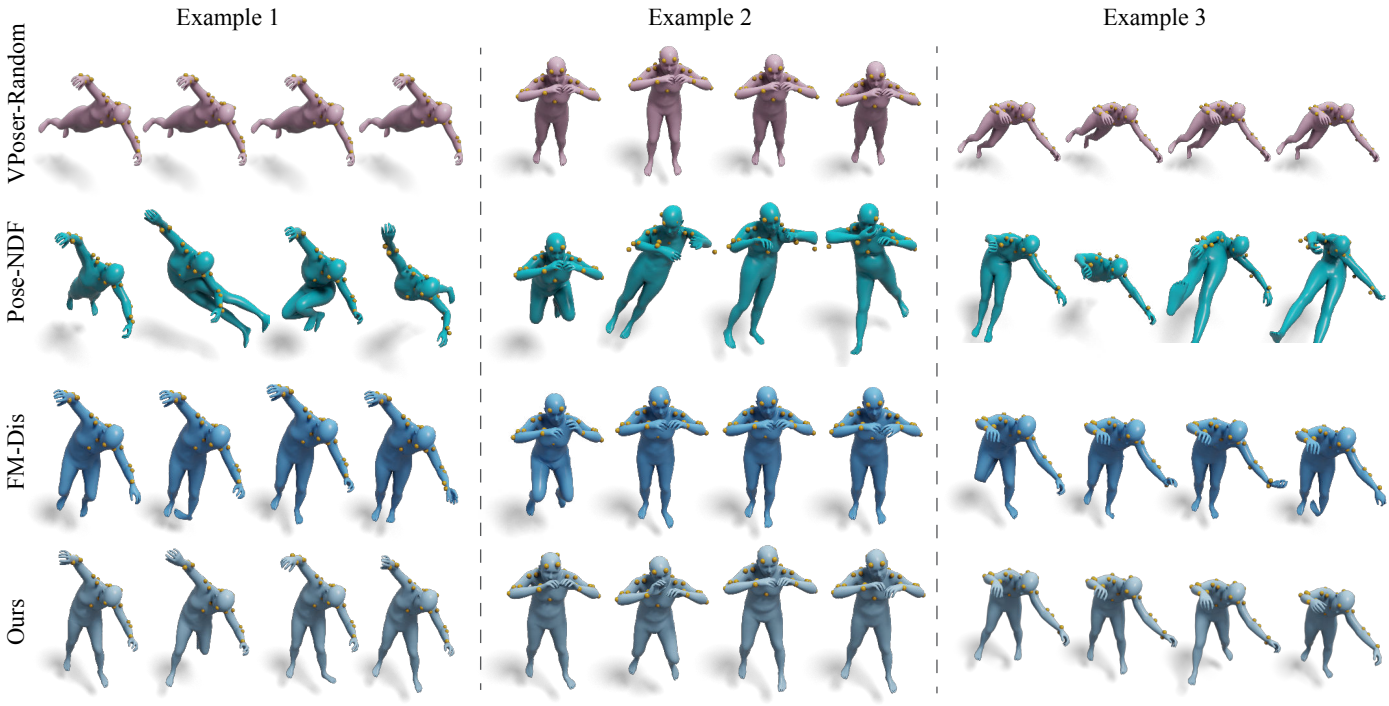

Figure 11. **IK Solver for occluded legs:** Given partial observation, where only one leg is occluded, we perform 3D pose completion. We observe that VPoser [51] based optimization generates realistic, yet very limited diversity in poses or almost similar poses. Pose-NDF [63] results unrealistic poses. FM-Dis generates almost similar poses and has no diversity. NRDF generates diverse and realistic poses as compared to the aforementioned pose priors.

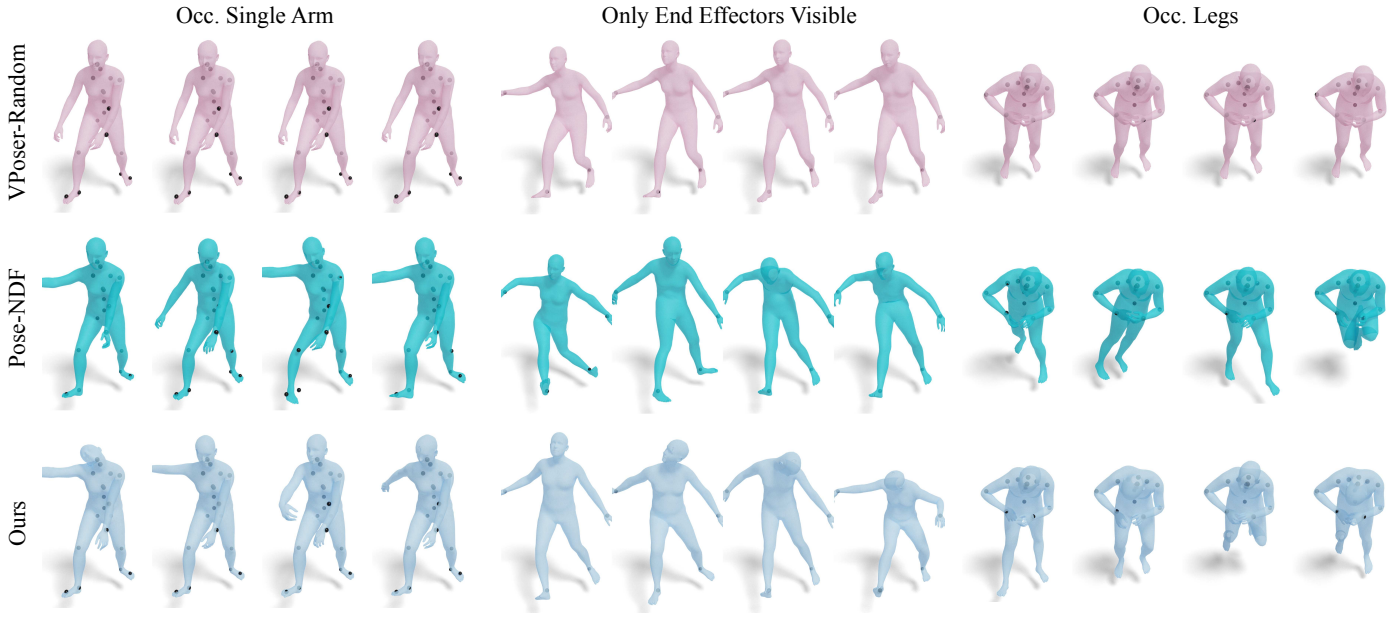

Figure 12. **IK Solver from partial/sparse joints:** Given partial observation (black joints), we perform 3D pose completion. We observe that VPoser [51] based optimization generates realistic, yet less diverse poses. Pose-NDF [63] generates more diverse, but sometimes unrealistic poses, especially in case of very sparse observations. NRDF generates diverse and realistic poses in all setups.

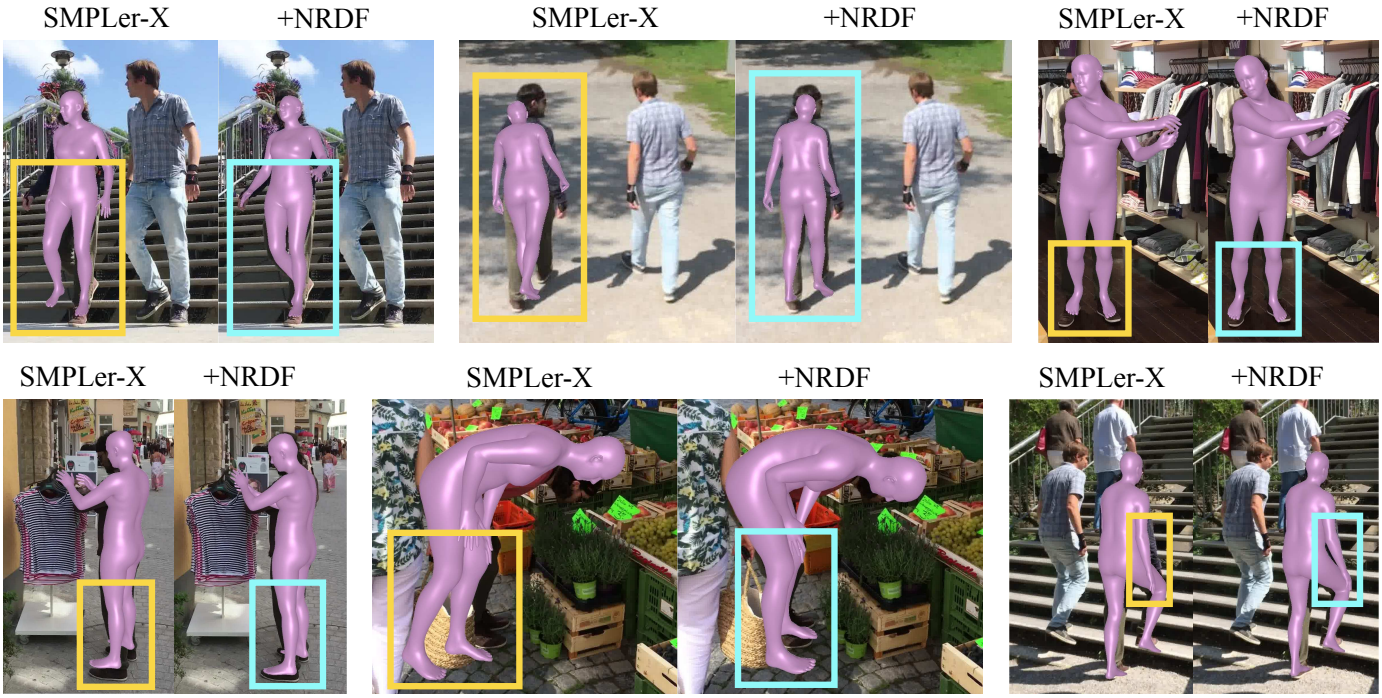

Figure 13. **3D pose and shape estimation from images:** (Top): We refine the results of SMPLer-X [18] network prediction using NRDF based optimization pipeline.