A. Preliminary: Diffusion Model

Following the notation of [16, 8, 1], we provide a detailed review of the diffusion models, which consist of a diffusion process and a reverse process.

Diffusion process. The diffusion process q generates corrupted samples $\boldsymbol{y}_1$, $\boldsymbol{y}_2$, ..., $\boldsymbol{y}_T$ by adding different levels of Gaussian noise to the original signal $\boldsymbol{y}_0$ at each timestep $t \in [0, T]$, which can be formulated as

$$\begin{aligned} q(\boldsymbol{y}_{1:T}|\boldsymbol{y}_0) &\coloneqq \prod_{t=1}^{T} q(\boldsymbol{y}_t|\boldsymbol{y}_{t-1}) \\ q(\boldsymbol{y}_t|\boldsymbol{y}_{t-1}) &\coloneqq \mathcal{N}(\boldsymbol{y}_t; \sqrt{1-\beta_t}\boldsymbol{y}_{t-1}, \beta_t \boldsymbol{I}) \end{aligned} \tag{1}$$

where β_t is the noise variance schedule [16] and $\boldsymbol{I}$ denotes the identity matrix. Following Ho *et al.* [8], we can rewrite Eq. 1 in a form that requires no iteration to yield $\boldsymbol{y}_t$ from $\boldsymbol{y}_0$:

$$\begin{aligned} q(\boldsymbol{y}_t|\boldsymbol{y}_0) &\coloneqq \mathcal{N}(\boldsymbol{y}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{y}_0, (1-\bar{\alpha}_t)\boldsymbol{I}) \\ &\coloneqq \sqrt{\bar{\alpha}_t}\boldsymbol{y}_0 + \epsilon\sqrt{1-\bar{\alpha}_t}, \epsilon \sim \mathcal{N}(0, \boldsymbol{I}) \end{aligned} \tag{2}$$

where $\bar{\alpha}_t \coloneqq \prod_{s=0}^{t} \alpha_s$ and $\alpha_t \coloneqq 1-\beta_t$. ϵ denotes the Gaussian noise. With a sufficiently large T and reasonable noise schedule β_t, the distribution of $q(\boldsymbol{y}_T)$ is nearly an isotropic Gaussian distribution $\mathcal{N}(0, \boldsymbol{I})$.

Reverse process. There are two ways to implement the reverse process p using a neural network parameterized by θ. One is by iterating $p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t)$ multiple times until $\boldsymbol{y}_0$, and the other is by getting $\boldsymbol{y}_0$ in one step directly from $p_\theta(\boldsymbol{y}_0|\boldsymbol{y}_t)$.

In the first case, it is found that the posterior $q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0)$ can be formulated as a Gaussian distribution as well using the Bayes' theorem:

$$q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0) = \mathcal{N}(\boldsymbol{y}_{t-1}; \tilde{\mu}(\boldsymbol{y}_t, \boldsymbol{y}_0), \tilde{\beta}_t \boldsymbol{I}) \tag{3}$$

where

$$\begin{aligned} \tilde{\beta}_t &\coloneqq \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \\ \tilde{\mu}_t(\boldsymbol{y}_t, \boldsymbol{y}_0) &\coloneqq \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\boldsymbol{y}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{y}_t \end{aligned} \tag{4}$$

are the variance and mean of the Gaussian distribution, respectively. However, Eq. 3 depends on $\boldsymbol{y}_0$ to measure the posterior, which is unknown in advance. Instead, we approximate $q(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t, \boldsymbol{y}_0)$ through

$$p_\theta(\boldsymbol{y}_{t-1}|\boldsymbol{y}_t) \coloneqq \mathcal{N}(\boldsymbol{y}_{t-1}; \mu_\theta(\boldsymbol{y}_t, t), \Sigma_\theta(\boldsymbol{y}_t, t)) \tag{5}$$

where

$$\begin{aligned} \Sigma_\theta(\boldsymbol{y}_t, t) &= \tilde{\beta}_t \boldsymbol{I} \\ \mu_\theta(\boldsymbol{y}_t, t) &= \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{y}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\boldsymbol{y}_t, t)\right) \end{aligned} \tag{6}$$

where $\epsilon_\theta(\boldsymbol{y}_t, t)$ is the noise predicted by the neural network, which is supervised by

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{y}_t, t, \epsilon}[||\epsilon_\theta(\boldsymbol{y}_t, t) - \epsilon||^2] \tag{7}$$

where $\epsilon \sim \mathcal{N}(0, \boldsymbol{I})$.

In the second case, we can directly predict the clean data from a learned network $f_\theta(\boldsymbol{y}_t, t)$, which is supervised by

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{y}_t, t}[||f_\theta(\boldsymbol{y}_t, t) - \boldsymbol{y}_0||^2] \tag{8}$$

The difference between the above two cases lies in the prediction target of the network, which in the first case is the noise ϵ at each timestep, and in the second case is the original signal $\boldsymbol{y}_0$. We choose to predict $\boldsymbol{y}_0$ in this work.

B. Preliminary: Projective Geometry

We refer to the process of a real-world object being photographed by a camera as the first *projection*. The obtained image is passed through the 3D pose estimator to predict multiple plausible 3D hypotheses. These hypotheses are projected a second time to the camera plane by JPMA, which is called *reprojection*. We introduce the reprojection function $\mathcal{P}(\cdot)$ in JPMA under two camera models: pinhole camera and distorted pinhole camera.

Pinhole camera. We denote the 3D coordinate of a human joint in the camera coordinate system as (X, Y, Z), and its reprojection to the camera plane as a 2D coordinate (u, v). Then, the reprojection under a pinhole camera (with 4 intrinsic camera parameters) can be formulated by a perspective transformation:

$$\begin{aligned} X' &= X/Z, \quad Y' = Y/Z \\ u &= f_x \cdot X' + c_x \\ v &= f_y \cdot Y' + c_y \end{aligned} \tag{9}$$

where f_x, f_y are the focal lengths expressed in pixel units and (c_x, c_y) is the principal point that is usually at the image center.

Distorted pinhole camera. Real lenses will bring distortion to the ideal pinhole camera model, resulting in a distorted pinhole camera (with 9 intrinsic camera parameters). The distortion can be categorized into two types: 1) radial distortion, which is caused by flawed radial curvature of a lens and can be approximated by three parameters k_1, k_2, k_3; 2) tangential distortion, which arises mainly from the tilt of a lens with respect to the image sensor array and can be approximated by two parameters p_1, p_2. When these two types of distortion are applied, Eq. 9 is extended as:

$$
\begin{aligned}
X' &= X/Z, \quad Y' = Y/Z \\
X_\text{d} &= X' \cdot (d_\text{r} + d_\text{t}) + p_1 r^2 \\
Y_\text{d} &= Y' \cdot (d_\text{r} + d_\text{t}) + p_2 r^2 \\
u &= f_x \cdot X_\text{d} + c_x \\
v &= f_y \cdot Y_\text{d} + c_y
\end{aligned}
\tag{10}
$$

where $d_\text{r} = 1 + k_1 r^2 + k_2 r^4 + k_3 r^6, r = \sqrt{X'^2 + Y'^2}$ and $d_\text{t} = 2p_1 X'^2 + 2p_2 Y'^2$, following the formulation in [4].

For calibrated cameras, we use the ground truth intrinsic camera parameters for JPMA. Otherwise, we use a network to estimate the parameters, which is discussed in Section D.2.

C. Algorithm

Algorithm 1 provides the pseudo-code of D3DP training procedure. First, we scale the ground truth 3D pose $\boldsymbol{y}_0$ to control the signal-to-noise ratio. Then, a diffusion process is constructed from the scaled signal to the noisy 3D pose. Next, we train a denoiser to reverse this process by using the noisy 3D pose, 2D keypoints, and the timestep to predict the clean 3D pose $\widetilde{\boldsymbol{y}}_0$. The entire framework is supervised by an MSE loss: $\mathcal{L} = ||\boldsymbol{y}_0 - \widetilde{\boldsymbol{y}}_0||_2$. By using this simple optimization objective, we avoid issues of sensitivity to hyperparameters (*e.g.*, balance factor between different loss functions) and training instability encountered in previous methods [24, 13].

Algorithm 2 provides the pseudo-code of D3DP inference procedure. We start by sampling H initial 3D poses from a Gaussian distribution and use the number of iterations K to determine the timestep for each iteration. In each iteration, noisy 3D poses are sent into the denoiser to estimate the uncontaminated 3D poses, which are used by DDIM to derive noisy inputs for the denoiser in the next iteration.

In addition, we combine D3DP with a common data augmentation scheme (*i.e.*, horizontal flipping) in deterministic 3D pose estimation [30, 19, 28, 18], and propose *diffusion-flipping*. Specifically, we horizontally flip input 2D keypoints as well as noisy 3D poses. The denoiser is then employed to produce the flipped predictions, which are flipped back and averaged with the original predictions to obtain the final 3D poses. The above steps are repeated in each iteration. Compared with previous approaches, the proposed method extends the data augmentation from once to K times, which reduces the cumulative error and increases the accuracy of predictions.

Algorithm 1 D3DP Training

```
def training_loss(2dp, 3dp_gt, T):
  # 2dp: [B, N, J, 2], 3dp_gt: [B, N, J, 3]
  # T: maximum number of timesteps
  # B: batch size, N: frame count, J: joint count

  # Signal scaling
  3dp_gt = 3dp_gt * scale

  # Corrupt 3dp_gt
  t = randint(0, T)                 # timestep
  noise = normal(mean=0, std=1)     # noise: [B, N, J, 3]
  alpha_cp = alpha_cumprod(t)
  3dp_crpt = sqrt(    alpha_cp) * 3dp_gt +
             sqrt(1 - alpha_cp) * noise

  # Denoise using a 3d pose estimator as backbone
  3dp_pred = denoiser(3dp_crpt, 2dp, t)

  # Set regression loss
  loss = MSE(3dp_pred, 3dp_gt)

  return loss
```

alpha_cumprod(t): cumulative product of α_i, *i.e.*, $\prod_{i=1}^{t} \alpha_i$

Algorithm 2 D3DP Inference

```
def inference(2dp, T, K, H):
  # 2dp: [B, N, J, 2], T: maximum number of timesteps
  # K, H: number of iterations and hypotheses

  # Initialize noisy 3d poses: [B, H, N, J, 3]
  3dp_t = normal(mean=0, std=1)

  # Sample timesteps uniformly
  times = reversed(linespace(0, T, K + 1))

  # [(T*(1-k/K), T*(1-(k+1)/K))], k = 0,...,K-1
  time_pairs = list(zip(times[:-1], times[1:])

  for t_now, t_next in zip(time_pairs):
    # Predict 3dp_0 from 3dp_t
    3dp_0 = denoiser(3dp_t, 2dp, t_now)

    # Diffusion flipping
    # Data augmentation using horizontal flipping
    if augment:
      2dp_hf = horiz_flipping(2dp)
      3dp_t_hf = horiz_flipping(3dp_t)
      3dp_0_hf = denoiser(3dp_t_hf, 2dp_hf, t_now)
      3dp_0 = (3dp_0 + horiz_flipping(3dp_0_hf)) / 2

    # Estimate 3dp_t at t_next
    3dp_t = ddim_step(3dp_t, 3dp_0, t_now, t_next)

  return 3dp_0
```

linespace: generate evenly spaced values

D. Additional Ablation Study

D.1. Components of D3DP

We conduct more ablation experiments on Human3.6M to study the design of D3DP in detail.

Regression target. We implement two alternatives: predicting the noise ϵ_t at each timestep of the reverse process or predicting the original 3D data $\boldsymbol{y}_0$. As shown in Table 1a, the latter achieves better results.

Note that the difference between our work and other concurrent diffusion-based methods [9, 7, 5] mainly lies in the regression target. The regression target of our model

Target	MPJPE↓
noise	40.2
original data	40.0

(a) Regression target.

Location	MPJPE↓
none	40.8
first layer	40.0
all layers	40.0

(b) Location of the added timestep embedding.

Type	MPJPE↓
none	40.6
flipping-once	40.3
diffusion-flipping	40.0

(c) Data augmentation.

Type	Method	MPJPE↓
IF	concat	40.0
EF	concat	40.9
EF	add	41.2
EF	CA	41.1

(d) 2D conditioning. IF: input fusion. EF: embedding fusion. CA: cross attention.

T	MPJPE↓
100	40.8
500	40.2
1000	40.0
2000	40.3

(e) Maximum number of timesteps.

Table 1: Ablation experiments of D3DP on Human3.6M to justify the selection of each component. H=1, K=1. Default settings are shown in gray .

Iteration	1	2	3	4	5
noise ϵ_t	48.6	43.5	41.0	40.5	40.0
original data $\boldsymbol{y}_0$	**39.9**	**39.9**	**39.8**	**39.8**	**39.7**

Table 2: Performance of two regression targets after each iteration. MPJPE↓ is reported. H=K=5. J-Agg is used.

is the original 3D data $\boldsymbol{y}_0$, while theirs is the noise ϵ_t at each timestep. Table 1a shows our method outperforms theirs. Further experiments (Table 2) reveal that predicting $\boldsymbol{y}_0$ yields good performance even in early iterations, while predicting ϵ_t does not. This is because in early iterations, when the input $\boldsymbol{y}_t$ is extremely noisy, it is more effective to predict the original signal $\boldsymbol{y}_0$ directly than to obtain $\boldsymbol{y}_0$ by predicting the noise ϵ_t and then subtracting it from $\boldsymbol{y}_t$. This property is valuable for real-time processing. For example, when K is fixed and computational resources are inadequate, the algorithm is required to produce predictions after the first iteration. Our method of predicting $\boldsymbol{y}_0$ still achieves satisfactory results, while theirs of predicting ϵ_t does not.

Location of the timestep embedding. We add the timestep embedding to the network in a similar way as the positional embedding [22]. Table 1b shows that adding it to the first layer of the network performs the same as all layers, hence the former is chosen for simplicity. Experimental results demonstrate that timestep embedding is crucial to the denoising process.

Data augmentation. Three data augmentation approaches are compared in Table 1c, including 1) no augmentation; 2) flipping-once, which flips the input, conducts denoising for K times, and flips the prediction again. The flipped prediction is then averaged with the unflipped prediction in the original branch to yield the final output; 3) diffusion-flipping, which applies the *flip-denoise-flip* process to each timestep (K times). The detailed architectures of these three approaches are shown in Fig. 1. Our diffusion-flipping achieves the best results because it averages the 3D poses of the original and flipped branches at each timestep, preventing the accumulation of errors. Other concurrent diffusion-based methods [9, 7, 5] don't use any augmentation or use

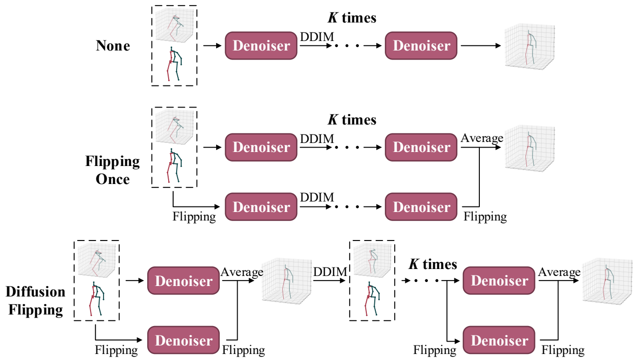

Figure 1: Detailed architectures of three data augmentation approaches.

the flipping-once method. Therefore, our model is more effective than theirs.

2D conditioning. As shown in Table 1d, we evaluate multiple fusion methods (concatenation, addition, and cross attention [22]) in two fusion types (input fusion and embedding fusion). For embedding fusion, two additional spatio-temporal Transformer layers are used to extract 2D features, after which these features are fused into the denoiser. The best fusion approach is concatenating noisy 3D poses and 2D conditions on the input side, which provides a fast and effective way to modify existing 3D pose estimators to fit the diffusion framework.

Maximum number of timesteps. Table 1e indicates that the best performance can be achieved by setting an appropriate maximum number of timesteps. When T is too small, we cannot diffuse the ground truth 3D poses to a Gaussian distribution during training, so the denoiser has trouble recovering a clean pose from Gaussian noise during inference. When T is too large, excessive samples become pure noise after diffusion. Then, the training process of the denoiser is affected and the denoiser cannot generalize well to 3D poses with varying levels of noise (*i.e.*, 3D poses at different timesteps) during inference.

Compatibility. The denoiser has the compatibility to use existing 3D human pose estimators as the backbone. We

Backbone	P-STMO [19]	PoseFormer [30]	STE [14]	TCN [18]
w/o diffusion	43.7	47.6	44.6	46.4
w/ diffusion	**42.2**	**45.7**	**43.8**	**44.5**

Table 3: Performance using other 3D estimators as backbone. MPJPE↓ is reported. H=K=1.

Camera Model	Access	MPJPE↓
w/ distortion	GT	39.74
w/o distortion	GT	39.80
w/ distortion	estimated	39.78
w/o distortion	estimated	39.82

Table 4: Impact of different camera models and different ways of accessing intrinsic camera parameters. H=5, K=5. GT: ground truth. J-Agg setting is used.

run our pipeline on a few other 3D estimators [19, 30, 14, 18]. Table 3 shows that our approach achieves performance gains over different backbone networks, which verifies the compatibility and versatility.

D.2. Intrinsic Camera Parameters in JPMA

As shown in Table 4, we investigate the effect of two factors on the performance of JPMA: 1) the camera model, including pinhole camera (w/o distortion) or distorted pinhole camera (w/ distortion); 2) the way of accessing the intrinsic camera parameters, including using the ground truth (GT) or using a 2-layer MLPs to estimate the parameters (estimated). When the ground truth intrinsic camera parameters are used, the ideal pinhole camera model without distortion shows a performance degradation of 0.06mm compared with the case with distortion. When a neural network is utilized to estimate the parameters (the ground truth is not available), the performance drops by only 0.04mm and 0.02mm in distorted and distortion-free cases, respectively. These results indicate that the proposed JPMA method is robust to the noise caused by incorrect camera models or inaccurate estimations of intrinsic camera parameters.

D.3. Variance of Hypotheses

Fig. 2 shows how the performance of the best, worst, and average hypotheses changes with the number of hypotheses H. As H increases, the performance of the average hypothesis remains essentially unchanged, with the worst hypothesis getting worse and the best hypothesis getting better. This result validates the statement in the main paper that the mean of all hypotheses remains roughly the same (the error stays around 39.9mm), while the variance rises.

E. Additional Quantitative Results

Table 5 provides quantitative comparisons between our D3DP with JPMA and the state-of-the-art approaches on Human3.6M when P-MPJPE is reported using 2D key-

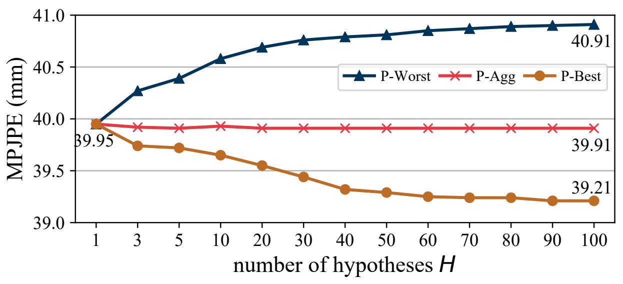

Figure 2: Effect of different numbers of hypotheses H on the performance of the best, worst, and average hypotheses. K=10.

points obtained from 2D detectors as inputs. Table 6 shows the results when MPJPE is reported using ground truth 2D keypoints as inputs. Without bells and whistles, D3DP transforms an existing deterministic 3D pose estimator into a probabilistic version with simple modifications and achieves considerable performance gains. Our method produces favorable results under conventional pose-level settings (P-Agg and P-Best), and the performance is further enhanced under the proposed joint-level settings (J-Agg and J-Best), which demonstrates the effectiveness of disentangling the hypothesis at the joint level. Experimental results show that the proposed method surpasses the others by a wide margin.

F. Additional Qualitative Results

F.1. Different Numbers of Hypotheses and Iterations

We show the qualitative results under different numbers of hypotheses H and iterations K in Fig. 3. When H increases (first row), the mean of hypotheses is basically unchanged while the variance increases, which is consistent with the conclusion in Section D.3. When K increases (second row), the variance also raises gradually, meaning that the diversity of hypotheses is improved. This is because DDIM re-adds different noise to the predicted 3D poses to generate inputs for the next iteration, resulting in a progressive increase in the gap between different hypotheses as K grows. When the variance is small, the 2D reprojections of all hypotheses are close to each other, which may affect the performance of JPMA (middle left: the solid red line is not better than the green one in the left foot region). When H, K are fixed (third row), we show the results of the 3D hypotheses estimated in each iteration. As k increases, the diversity of hypotheses is improved and the advantage of JPMA over average becomes apparent.

F.2. In-the-Wild Videos

We train our method on Human3.6M dataset and evaluate on in-the-wild videos from 3DPW [23], Penn Ac-

Deterministic Methods

P-MPJPE		Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Pur.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg
TCN [18] (N=243)*	CVPR'19	34.1	36.1	34.4	37.2	36.4	42.2	34.4	33.6	45.0	52.5	37.4	33.8	37.8	25.6	27.3	36.5
RIE [20] (N=243)*	MM'21	32.5	36.2	33.2	35.3	35.6	42.1	32.6	31.9	42.6	47.9	36.6	32.1	34.8	24.2	25.8	35.0
Anatomy [2] (N=243)*	TCSVT'21	32.6	35.1	32.8	35.4	36.3	40.4	32.4	32.3	42.7	49.0	36.8	32.4	36.0	24.9	26.5	35.0
PoseFormer [30] (N=81)*	ICCV'21	32.5	34.8	32.6	34.6	35.3	39.5	32.1	32.0	42.8	48.5	34.8	32.4	35.3	24.5	36.0	34.6
U-CDGCN [10] (N=96)*	MM'21	29.8	34.4	31.9	31.5	35.1	40.0	30.3	30.8	42.6	49.0	35.9	31.8	35.0	25.7	23.6	33.8
STE [14] (N=351)*	TMM'22	32.7	35.5	32.5	35.4	35.9	41.6	33.0	31.9	45.1	50.1	36.3	33.5	35.1	23.9	25.0	35.2
P-STMO [19] (N=243)*	ECCV'22	31.3	35.2	32.9	33.9	35.4	39.3	32.5	31.5	44.6	48.2	36.3	32.9	34.4	23.8	23.9	34.4
MixSTE [28] (N=243)*	CVPR'22	30.8	33.1	30.3	31.8	33.1	39.1	31.1	30.5	42.5	44.5	34.0	30.8	32.7	22.1	22.9	32.6
MixSTE [28] (N=243)*‡	CVPR'22	30.8	32.7	30.6	31.9	33.1	38.6	30.8	30.4	42.4	46.4	34.2	30.7	32.3	21.8	22.6	32.6
DUE [27] (N=300)*	MM'22	30.3	34.6	29.6	31.7	31.6	38.9	31.8	31.9	39.2	42.8	32.1	32.6	31.4	25.1	23.8	32.5
D3DP (N=243, H=1, K=1)*		30.6	32.5	29.1	31.0	31.9	37.6	30.3	29.4	40.6	43.6	33.3	30.5	31.4	21.5	22.4	31.7

Probabilistic Methods

P-MPJPE		Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Pur.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg
CVAE [21] (N=1, H=200, P-Agg)	ICCV'19	35.3	35.9	45.8	42.0	40.9	52.6	36.9	35.8	43.5	51.9	44.3	38.8	45.5	29.4	34.3	40.9
GAN [13] (N=1, H=10, P-Agg)	BMVC'20	42.1	44.7	45.4	51.0	49.3	51.5	41.2	46.2	57.5	70.8	48.7	44.1	50.8	42.1	43.7	48.7
GraphMDN [17] (N=1, H=5, P-Agg)	IJCNN'21	39.7	43.4	44.0	46.2	48.8	54.5	39.4	41.1	55.0	69.0	48.0	43.7	49.6	38.4	42.4	46.9
NF [24] (N=1, H=1, P-Agg)	ICCV'21	37.8	41.7	42.1	41.8	46.5	50.2	38.0	39.2	51.7	61.8	45.4	42.6	45.7	33.7	38.5	43.8
MHFormer [15] (N=351, H=3, P-Agg)*	CVPR'22	31.5	34.9	32.8	33.6	35.3	39.6	32.0	32.2	43.5	48.7	36.4	32.6	34.3	23.9	25.1	34.4
D3DP (N=243, H=1, K=1, P-Agg)*		30.6	32.5	29.1	31.0	31.9	37.6	30.3	29.4	40.6	43.6	33.3	30.5	31.4	21.5	22.4	31.7
D3DP (N=243, H=20, K=10, P-Agg)*		30.6	32.5	29.1	30.9	31.9	37.5	30.2	29.4	40.6	43.4	33.3	30.4	31.4	21.5	22.4	31.7
D3DP (N=243, H=20, K=10, J-Agg)*		30.6	32.4	29.2	30.9	31.9	37.4	30.2	29.3	40.4	43.2	33.2	30.4	31.3	21.5	22.3	31.6
MDN [12] (N=1, H=5, P-Best♯)	CVPR'19	35.5	39.8	41.3	42.3	46.0	48.9	36.9	37.3	51.0	60.6	44.9	40.2	44.1	33.1	36.9	42.6
CVAE [21] (N=1, H=200, P-Best♯)	ICCV'19	27.6	27.5	34.9	32.3	33.3	42.7	28.7	28.0	36.1	42.7	36.0	30.7	37.6	24.3	27.1	32.7
GAN [13] (N=1, H=10, P-Best♯)	BMVC'20	38.5	41.7	39.6	45.2	45.8	46.5	37.8	42.7	52.4	62.9	45.3	40.9	45.3	38.6	38.4	44.3
GraphMDN [17] (N=1, H=200, P-Best♯)	IJCNN'21	30.8	34.7	33.6	34.2	39.6	42.2	31.0	31.9	42.9	53.5	38.1	34.1	38.0	29.6	31.1	36.3
NF [24] (N=1, H=200, P-Best♯)	ICCV'21	27.9	31.4	29.7	30.2	34.9	37.1	27.3	28.2	39.0	46.1	34.2	32.3	33.6	26.1	27.5	32.4
D3DP (N=243, H=1, K=1, P-Best♯)*		30.6	32.5	29.1	31.0	31.9	37.6	30.3	29.4	40.6	43.6	33.3	30.5	31.4	21.5	22.4	31.7
D3DP (N=243, H=20, K=10, P-Best♯)*		30.2	32.1	28.8	30.4	31.5	37.0	29.8	28.9	39.9	42.4	32.8	30.0	30.9	21.3	22.1	31.2
D3DP (N=243, H=20, K=10, J-Best♯)*		27.5	29.4	26.6	27.7	29.2	34.3	27.5	26.2	37.3	39.0	30.3	27.7	28.2	19.6	20.3	28.7

Table 5: Results on Human3.6M in millimeters under P-MPJPE. N, H, K: the number of input frames, hypotheses, and iterations of the proposed D3DP. (‡) - Our implementation. (♯) - Not feasible in real-world applications. (*) - Use CPN [3] as the 2D keypoint detector to generate the inputs. Red: Best. Blue: Second best. Gray : our method.

Deterministic Methods

MPJPE		Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Pur.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg
TCN [18] (N=243)	CVPR'19	35.2	40.2	32.7	35.7	38.2	45.5	40.6	36.1	48.8	47.3	37.8	39.7	38.7	27.8	29.5	37.8
SRNet [25] (N=243)	ECCV'20	34.8	32.1	28.5	30.7	31.4	36.9	35.6	30.5	38.9	40.5	32.5	31.0	29.9	22.5	24.5	32.0
Anatomy [2] (N=243)	TCSVT'21	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	32.3
PoseFormer [30] (N=81)	ICCV'21	30.0	33.6	29.9	31.0	30.2	33.3	34.8	31.4	37.8	38.6	31.7	31.5	29.0	23.3	23.1	31.3
RIE [20] (N=243)	MM'21	29.5	30.8	28.8	29.1	30.7	35.2	31.7	27.8	34.5	36.0	30.3	29.4	28.9	24.1	24.7	30.1
Ray3D [26] (N=9)	CVPR'22	31.2	35.7	34.6	33.6	35.0	37.5	37.2	30.9	42.5	41.3	34.6	36.5	32.0	29.7	28.9	34.4
P-STMO [19] (N=243)	ECCV'22	28.5	30.1	28.6	27.9	29.8	33.2	31.3	27.8	36.0	37.4	29.7	29.5	28.1	21.0	21.0	29.3
STE [14] (N=351)	TMM'22	27.1	29.4	26.5	27.1	28.6	33.0	30.7	26.8	38.2	34.7	29.1	29.8	26.8	19.1	19.8	28.5
DUE [27] (N=300)	MM'22	22.1	23.1	20.1	22.7	21.3	24.1	23.6	21.6	26.3	24.8	21.7	21.4	21.8	16.7	18.7	22.0
MixSTE [28] (N=243)	CVPR'22	21.6	22.0	20.4	21.0	20.8	26.3	24.7	21.9	26.9	24.9	21.2	21.5	20.8	14.7	15.7	21.6
MixSTE [28] (N=243)‡	CVPR'22	22.9	21.7	21.0	21.4	20.8	23.5	24.1	21.8	25.3	23.5	21.4	20.1	19.5	15.3	16.6	21.3
D3DP (N=243, H=1, K=1)		19.9	19.6	19.7	19.3	20.2	22.7	21.5	19.2	25.5	24.0	20.1	18.9	19.0	14.0	14.5	19.9

Probabilistic Methods

MPJPE		Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Pur.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg
GraphMDN [17] (N=1, H=5, P-Agg)	IJCNN'21	33.9	39.9	33.0	35.4	36.8	44.4	38.9	33.0	41.0	50.0	36.4	38.3	37.8	28.2	31.5	37.2
MHFormer [15](N=351, H=3, P-Agg)	CVPR'22	27.7	32.1	29.1	28.9	30.0	33.9	33.0	31.2	37.0	39.3	30.3	31.0	29.4	22.2	23.0	30.5
D3DP (N=243, H=1, K=1, P-Agg)		19.9	19.6	19.7	19.3	20.2	22.7	21.5	19.2	25.5	24.0	20.1	18.9	19.0	14.0	14.5	19.9
D3DP (N=243, H=20, K=10, P-Agg)		19.9	19.5	19.6	19.2	20.1	22.4	21.5	19.1	25.4	23.7	20.0	18.9	18.8	14.0	14.5	19.8
D3DP (N=243, H=20, K=10, J-Agg)		19.9	19.4	19.4	19.0	19.8	22.0	21.4	19.1	24.8	23.2	19.6	18.7	18.6	14.0	14.5	19.6
GAN [13] (N=1, H=10, P-Best♯)	BMVC'20	54.8	61.9	48.6	63.6	55.8	73.7	59.0	61.3	62.2	85.7	52.8	60.2	57.5	51.3	56.8	60.0
GraphMDN [17] (N=1, H=5, P-Best♯)	IJCNN'21	28.9	34.5	28.2	30.2	31.5	38.5	32.3	28.6	35.7	43.3	31.9	32.1	33.3	25.2	27.8	31.8
D3DP (N=243, H=1, K=1, P-Best♯)		19.9	19.6	19.7	19.3	20.2	22.7	21.5	19.2	25.5	24.0	20.1	18.9	19.0	14.0	14.5	19.9
D3DP (N=243, H=20, K=10, P-Best♯)		19.7	19.3	19.2	18.8	19.6	21.9	21.2	18.6	24.7	23.0	19.4	18.4	18.4	13.9	14.4	19.4
D3DP (N=243, H=20, K=10, J-Best♯)		18.7	18.2	18.4	17.8	18.6	20.9	20.2	17.7	23.8	21.8	18.5	17.4	17.4	13.1	13.6	18.4

Table 6: Results on Human3.6M in millimeters under MPJPE. N, H, K: the number of input frames, hypotheses, and iterations of the proposed D3DP. (‡) - Our implementation. (♯) - Not feasible in real-world applications. The ground truth 2D keypoints are used as inputs. Red: Best. Blue: Second best. Gray : our method.

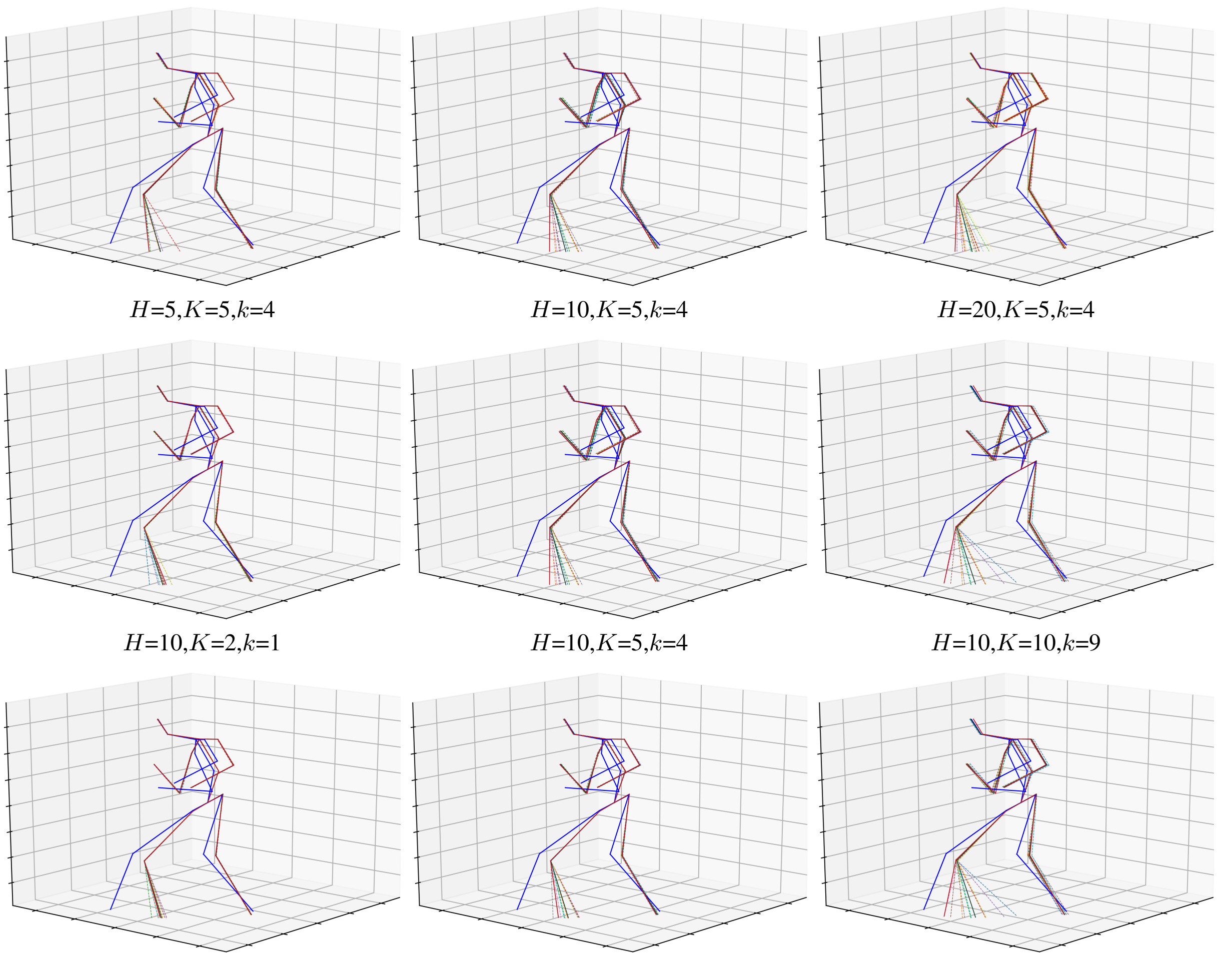

Figure 3: Qualitative results under different numbers of hypotheses H and iterations K. Top: H varies, when K=5. The results are the outputs of the last iteration, *i.e.*, k=4 for all three subfigures. Middle: K varies, when H=10. The results are the outputs of the last iteration, *i.e.*, k=1, 4, 9 for three subfigures respectively. Bottom: H=10, K=10. The results are the outputs of the first, intermediate, and last iterations, *i.e.*, k=0, 5, 9 for three subfigures respectively. Dashed line: predicted 3D pose hypotheses. Each color represents an individual hypothesis. Solid blue line: ground truth 3D poses. Solid red line: the final prediction obtained by using JPMA as the aggregation method. Solid green line: the final prediction obtained by using average as the aggregation method.

tion [29], JHMDB [11], and youtube. We use AlphaPose [6] as the 2D keypoint detector to generate 2D poses. As shown in Fig. 4, our method achieves satisfactory performance in most of the frames. For the cases of severe occlusion (3rd row), fast motion (5th row), low illumination (7th row), and rare poses (8th row), the proposed method generates highly uncertain predictions to represent the possible locations of 3D poses. Besides, our method can also generalize to animations (8th row) and monkey poses (9th row) because 2D keypoints are used as inputs, excluding the interference of complex textures. Theoretically, our method can be applied to any type of humanoid 3D pose estimation.

F.3. Analysis of Failure Cases

Our method may fail under the previously mentioned cases such as severe occlusions, fast motion, rare poses, *etc*. The 3rd row of Fig. 4 shows a person walking through the woods. Most of her body is occluded and our approach cannot obtain a very accurate result in this case. Besides, the person in the 6th row of Fig. 4 raises his hand above his head, a movement that is rare in the training set. Our method fails to generalize to this pose and therefore produces incorrect predictions.

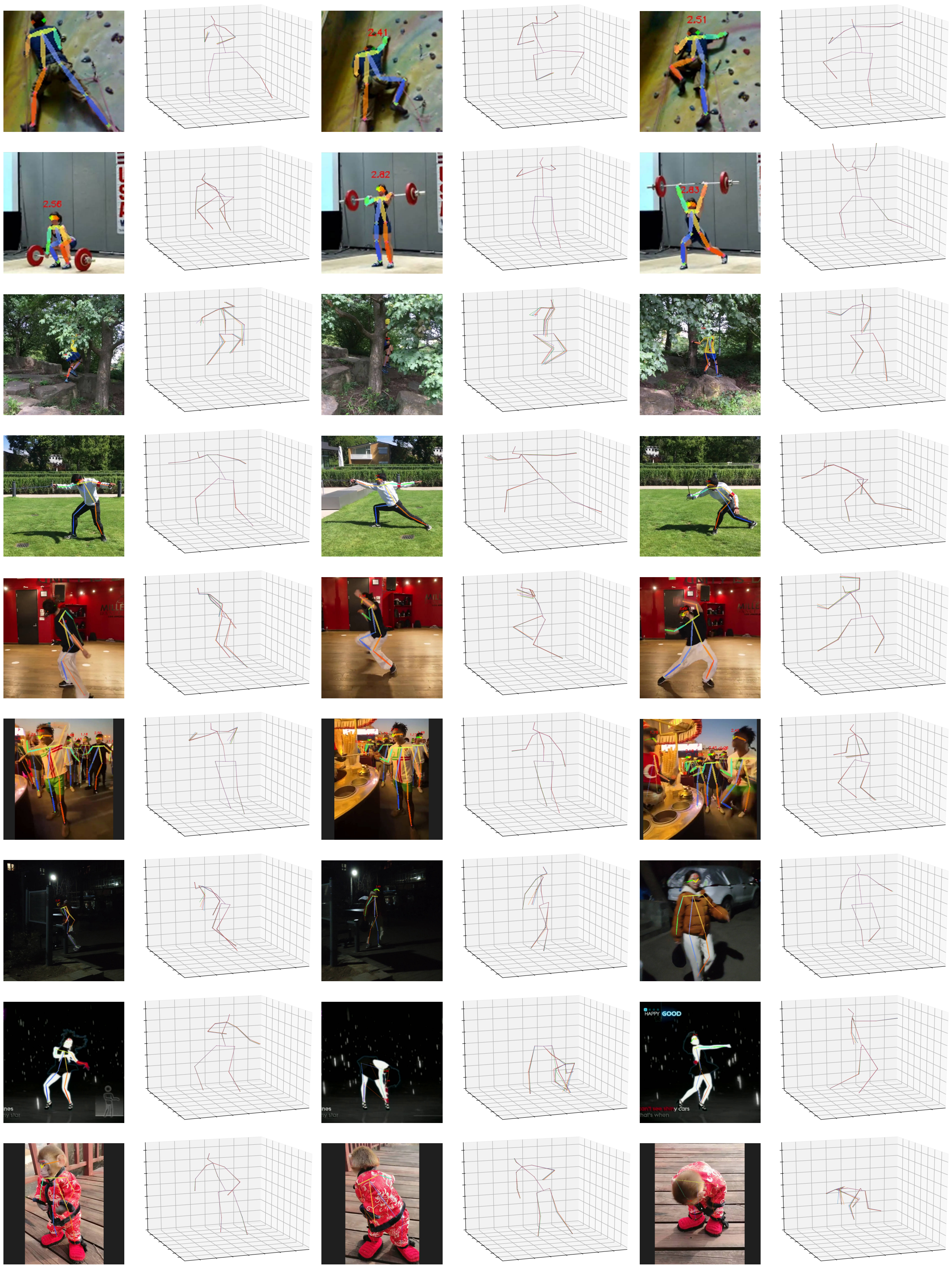

Figure 4: Qualitative results of the proposed method on in-the-wild videos. Dashed line: predicted 3D pose hypotheses. Each color represents an individual hypothesis. H=5, K=5

References

[1] Shoufa Chen, Peize Sun, Yibing Song, and Ping Luo. Diffusiondet: Diffusion model for object detection. *arXiv preprint arXiv:2211.09788*, 2022. 1

[2] Tianlang Chen, Chen Fang, Xiaohui Shen, Yiheng Zhu, Zhili Chen, and Jiebo Luo. Anatomy-aware 3d human pose estimation with bone-based pose decomposition. *IEEE Transactions on Circuits and Systems for Video Technology*, 2021. 5

[3] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7103–7112, 2018. 5

[4] Hanbyel Cho, Yooshin Cho, Jaemyung Yu, and Junmo Kim. Camera distortion-aware 3d human pose estimation in video with optimization-based meta-learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11169–11178, 2021. 2

[5] Jeongjun Choi, Dongseok Shim, and H Jin Kim. Diffupose: Monocular 3d human pose estimation via denoising diffusion probabilistic model. *arXiv preprint arXiv:2212.02796*, 2022. 2, 3

[6] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. Rmpe: Regional multi-person pose estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 2334–2343, 2017. 6

[7] Jia Gong, Lin Geng Foo, Zhipeng Fan, Qiuhong Ke, Hossein Rahmani, and Jun Liu. Diffpose: Toward more reliable 3d pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13041–13051, 2023. 2, 3

[8] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 1

[9] Karl Holmquist and Bastian Wandt. Diffpose: Multi-hypothesis human pose estimation using diffusion models. *arXiv preprint arXiv:2211.16487*, 2022. 2, 3

[10] Wenbo Hu, Changgong Zhang, Fangneng Zhan, Lei Zhang, and Tien-Tsin Wong. Conditional directed graph convolution for 3d human pose estimation. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 602–611, 2021. 5

[11] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *International Conf. on Computer Vision (ICCV)*, pages 3192–3199, Dec. 2013. 6

[12] Chen Li and Gim Hee Lee. Generating multiple hypotheses for 3d human pose estimation with mixture density network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9887–9895, 2019. 5

[13] Chen Li and Gim Hee Lee. Weakly supervised generative network for multiple 3d human pose hypotheses. *BMVC*, 2020. 2, 5

[14] Wenhao Li, Hong Liu, Runwei Ding, Mengyuan Liu, Pichao Wang, and Wenming Yang. Exploiting temporal contexts with strided transformer for 3d human pose estimation. *IEEE Transactions on Multimedia*, 2022. 4, 5

[15] Wenhao Li, Hong Liu, Hao Tang, Pichao Wang, and Luc Van Gool. Mhformer: Multi-hypothesis transformer for 3d human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13147–13156, 2022. 5

[16] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. 1

[17] Tuomas Oikarinen, Daniel Hannah, and Sohrob Kazerounian. Graphmdn: Leveraging graph structure and deep learning to solve inverse problems. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9. IEEE, 2021. 5

[18] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7753–7762, 2019. 2, 4, 5

[19] Wenkang Shan, Zhenhua Liu, Xinfeng Zhang, Shanshe Wang, Siwei Ma, and Wen Gao. P-stmo: Pre-trained spatial temporal many-to-one model for 3d human pose estimation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part V*, pages 461–478. Springer, 2022. 2, 4, 5

[20] Wenkang Shan, Haopeng Lu, Shanshe Wang, Xinfeng Zhang, and Wen Gao. Improving robustness and accuracy via relative information encoding in 3d human pose estimation. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 3446–3454, 2021. 5

[21] Saurabh Sharma, Pavan Teja Varigonda, Prashast Bindal, Abhishek Sharma, and Arjun Jain. Monocular 3d human pose estimation by generation and ordinal ranking. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2325–2334, 2019. 5

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 3

[23] Timo von Marcard, Roberto Henschel, Michael Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *European Conference on Computer Vision (ECCV)*, sep 2018. 4

[24] Tom Wehrbein, Marco Rudolph, Bodo Rosenhahn, and Bastian Wandt. Probabilistic monocular 3d human pose estimation with normalizing flows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11199–11208, 2021. 2, 5

[25] Ailing Zeng, Xiao Sun, Fuyang Huang, Minhao Liu, Qiang Xu, and Stephen Lin. Srnet: Improving generalization in 3d human pose estimation with a split-and-recombine approach. In *European Conference on Computer Vision*, pages 507–523. Springer, 2020. 5

[26] Yu Zhan, Fenghai Li, Renliang Weng, and Wongun Choi. Ray3d: ray-based 3d human pose estimation for monocular

absolute 3d localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13116–13125, 2022. 5

[27] Jinlu Zhang, Yujin Chen, and Zhigang Tu. Uncertainty-aware 3d human pose estimation from monocular video. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 5102–5113, 2022. 5

[28] Jinlu Zhang, Zhigang Tu, Jianyu Yang, Yujin Chen, and Junsong Yuan. Mixste: Seq2seq mixed spatio-temporal encoder for 3d human pose estimation in video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13232–13242, 2022. 2, 5

[29] Weiyu Zhang, Menglong Zhu, and Konstantinos G Derpanis. From actemes to action: A strongly-supervised representation for detailed action understanding. In *Proceedings of the IEEE international conference on computer vision*, pages 2248–2255, 2013. 6

[30] Ce Zheng, Sijie Zhu, Matias Mendieta, Taojiannan Yang, Chen Chen, and Zhengming Ding. 3d human pose estimation with spatial and temporal transformers. *arXiv preprint arXiv:2103.10455*, 2021. 2, 4, 5