Supplementary Material: Self-supervised 3D Human Pose Estimation from a Single Image

Jose Sosa, David Hogg
School of Computing, University of Leeds
{scjasm, D.C.Hogg}@leeds.ac.uk

Overview

We provide comparative per-activity quantitative results on Human3.6M (section 1); and additional qualitative results for Human3.6M [3] (section 2), MPI-INF-3DHP [9] (section 3), and HandDB [10] (section 4) datasets. Moreover, we include more details about the implementation and structure of the networks (section 5).

1. Quantitative Results on Human3.6M

Method	**Assumptions**	**Dir.**	**Disc.**	**Eat**	**Greet**	**Phon.**	**Photo**	**Pose**	**Purch.**	**Sit**	**SitD.**	**Smoke**	**Wait**	**Walk**	**WalkD**	**WalkT**	**Avg.(↓)**
Chen [1]	Full-3D	89.8	97.6	89.9	107.9	107.3	139.2	93.6	136.1	133.1	240.1	106.6	106.2	87.0	114.0	**90.6**	114.2
Kundu [7]	3D Kin.	**80.2**	81.3	**86.0**	**86.7**	**94.1**	**83.4**	87.5	**84.2**	**101.2**	**110.9**	**86.0**	87.8	**86.9**	**94.3**	90.9	89.4
Ours	Unp. 2D	84.4	**77.8**	89.0	99.2	100.6	101.8	**77.2**	86.5	112.2	144.4	97.3	**80.4**	93.6	103.3	102.5	96.7

Table 1. **Extended quantitative results on the Human3.6M dataset.** The P-MPJPE for each activity on the Human3.6M test set (subjects 9 and 11). The performance is compared with two state-of-the-art approaches for which per-activity data is available. Note that by only using unpaired 2D poses, we outperform methods that rely on paired 3D annotations [1]. We perform similarly; and even better for some activities (in bold) than methods relying on 3D kinematic constraints [7].

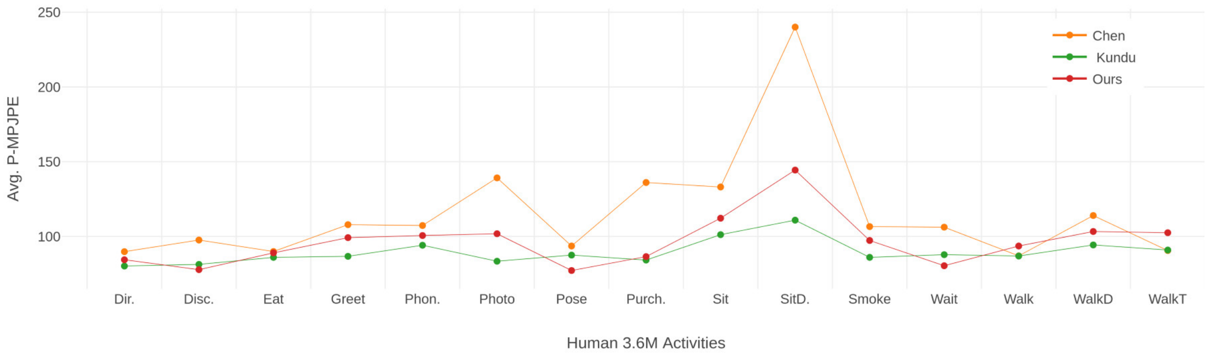

Figure 1. **Distribution of P-MPJPE scores for each activity on the Human3.6M dataset.** We achieve superior performance with only an unpaired prior on 2D poses than [1]. Our method also outperforms [7] (which assumes 3D kinematic constraints) in 20% of the activities.

2. Qualitative Results on Human3.6M

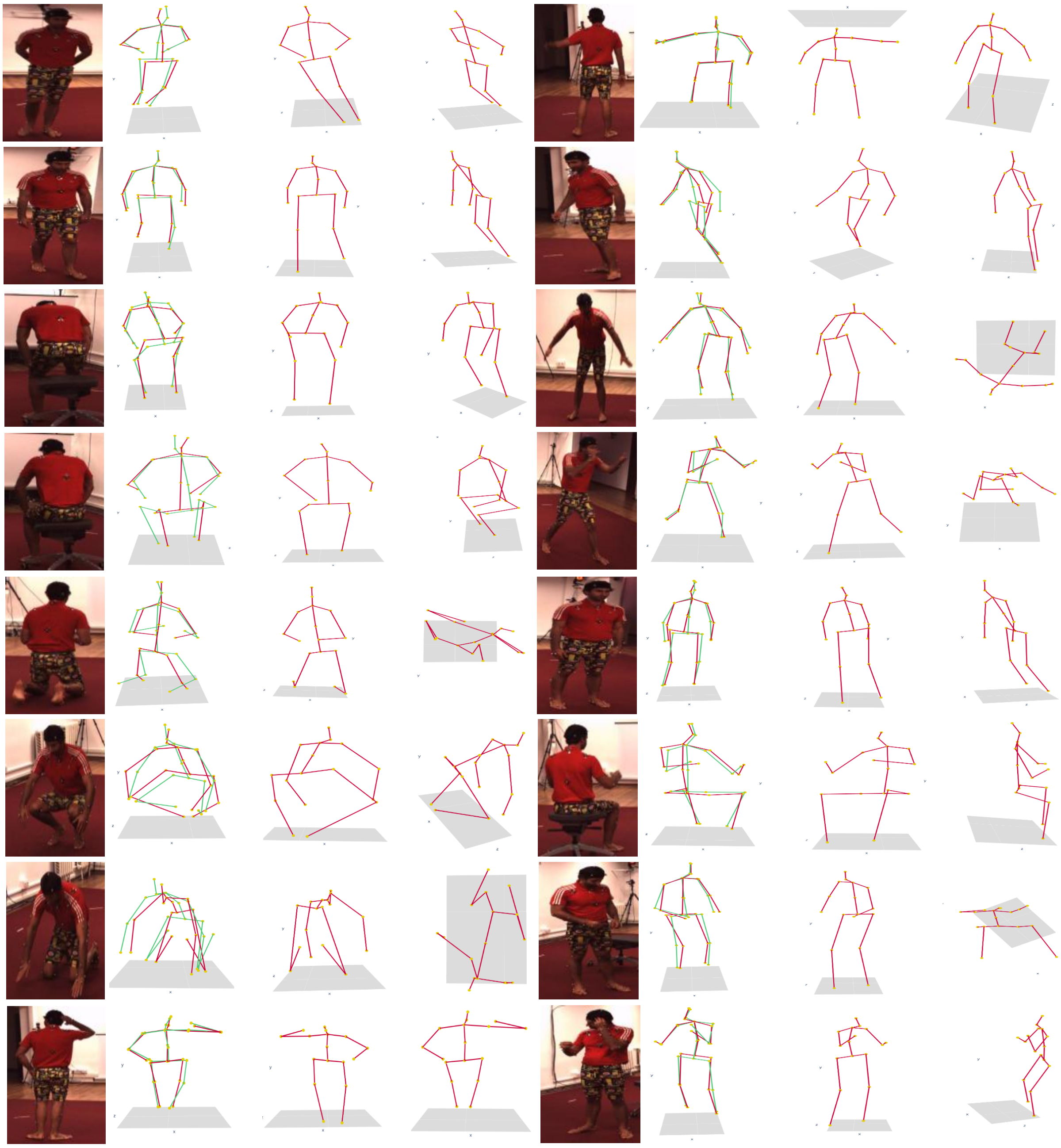

Figure 2. **3D pose predictions on images corresponding to subject 9 (S9) from Human3.6M dataset.** The first and fifth columns show the input image, and the following columns (second and sixth) display the actual 3D pose from the dataset (coloured in green) aligned with the 3D pose predicted by our model (coloured in red). The remaining columns show novel views of the predicted 3D pose.

Qualitative Results on Human3.6M (Continue)

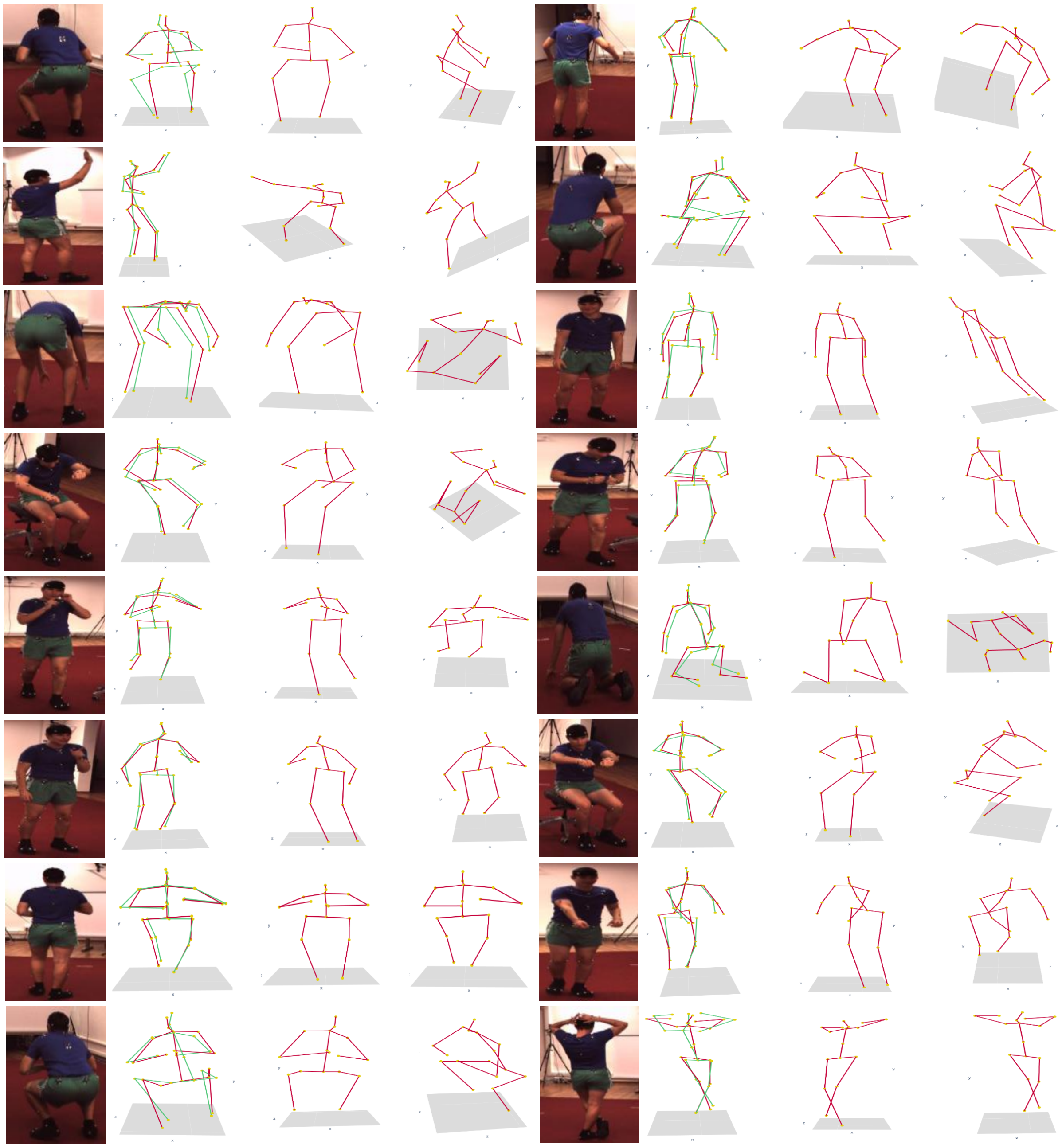

Figure 3. **3D pose predictions on images corresponding to subject 11 (S11) from Human3.6M dataset.** The first and fifth columns show the input image, and the following columns (second and sixth) display the actual 3D pose from the dataset (coloured in green) aligned with the 3D pose predicted by our model (coloured in red). The remaining columns show novel views of the predicted 3D pose.

3. Qualitative Results on MPI-INF-3DHP

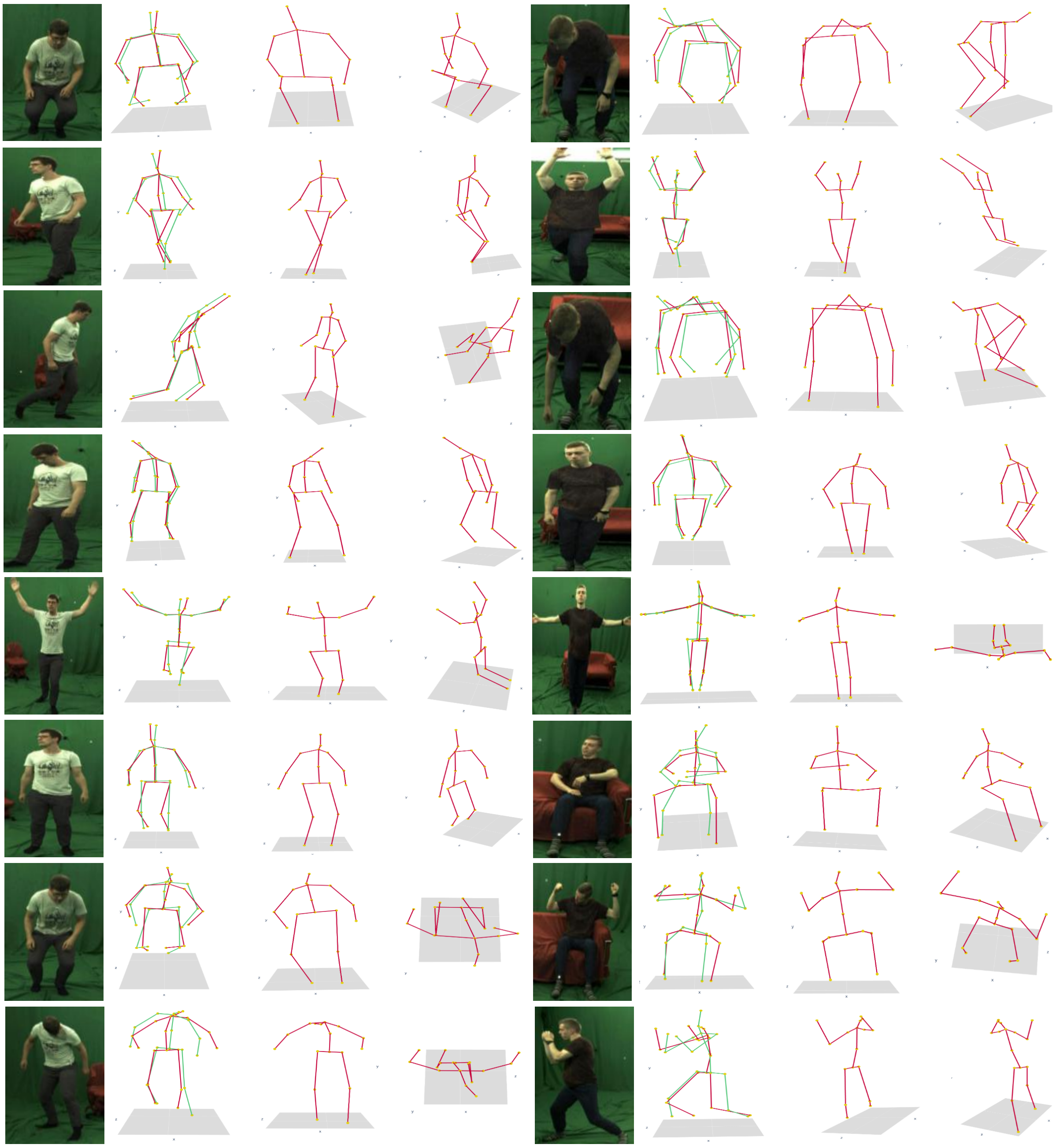

Figure 4. **3D pose predictions on images corresponding to subjects 1 and 2 from MPI-INF-3DHP dataset.** The first and fifth columns show the input image, and the following columns (second and sixth) display the actual 3D pose from the dataset (coloured in green) aligned with the 3D pose predicted by our model (coloured in red). The remaining columns show novel views of the predicted 3D pose.

Qualitative Results on MPI-INF-3DHP (Continue)

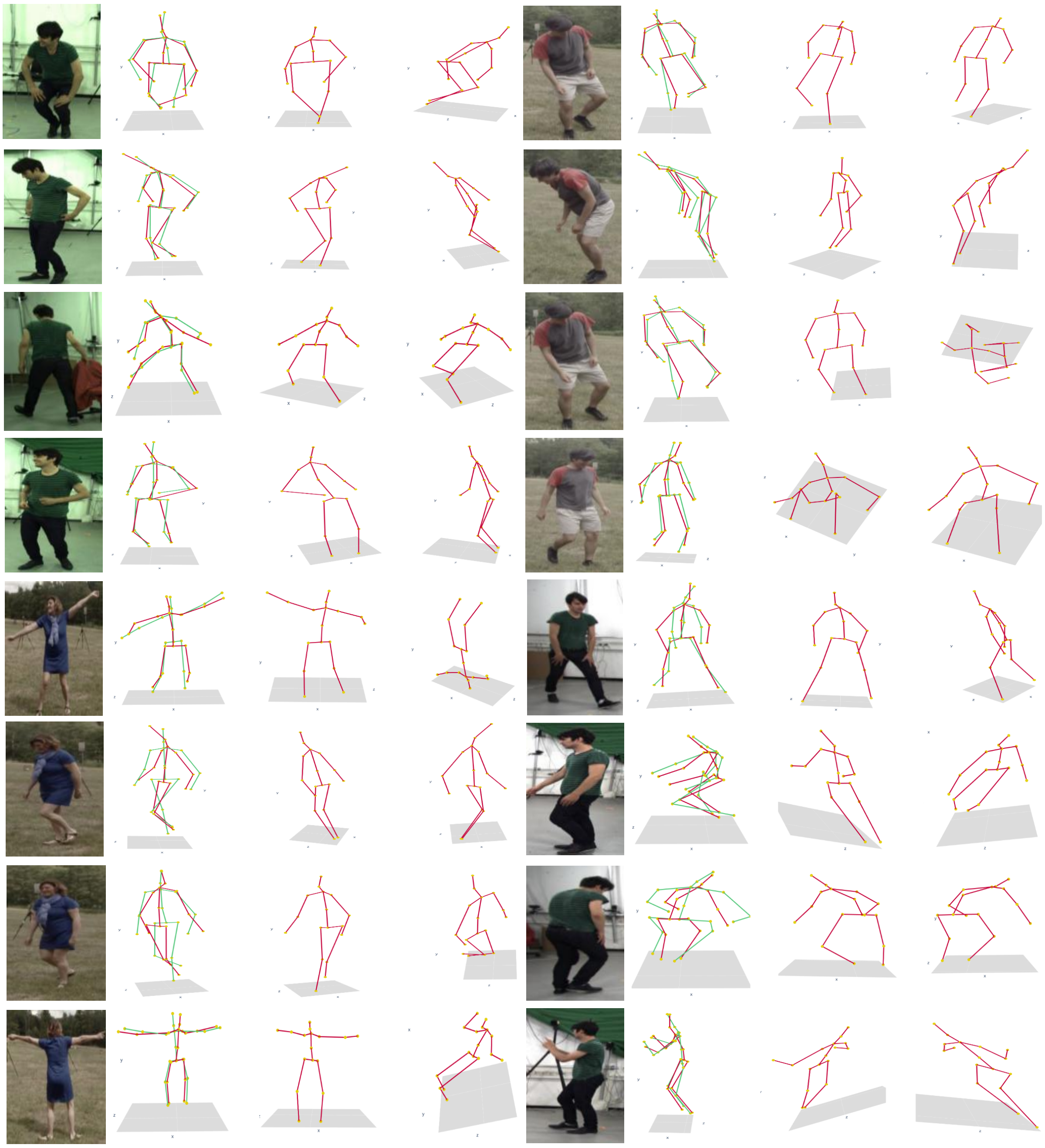

Figure 5. **3D pose predictions on images corresponding to subjects 3,4,5, and 6 from MPI-INF-3DHP dataset.** The first and fifth columns show the input image, and the following columns (second and sixth) display the actual 3D pose from the dataset (coloured in green) aligned with the 3D pose predicted by our model (coloured in red). The remaining columns show novel views of the predicted 3D pose.

4. Qualitative Results on HandDB

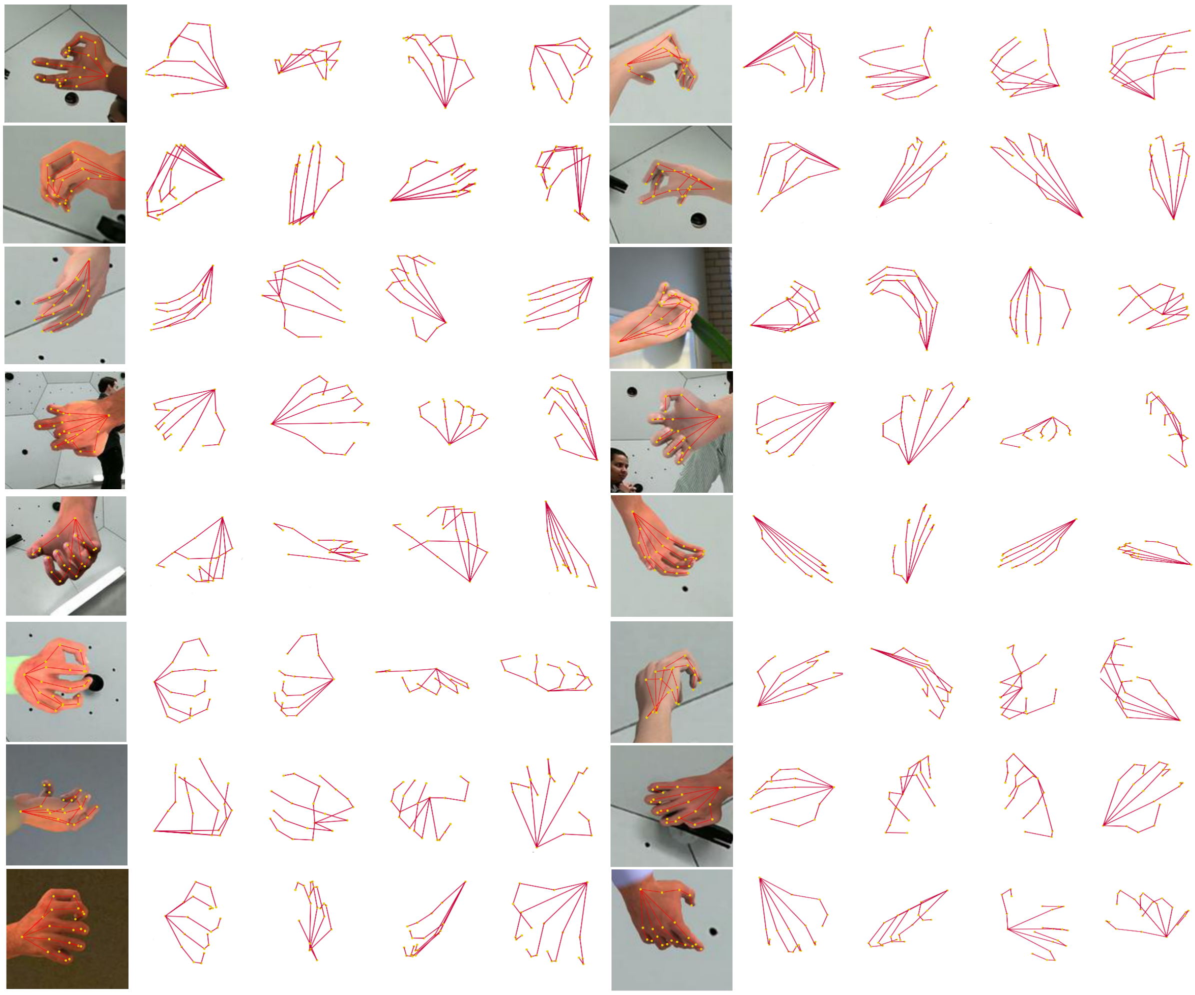

Figure 6. **3D hand pose predictions on synthetic hand images from HandDB dataset.** The first and sixth columns show the input image with its corresponding 2D ground-truth superimposed. The remaining columns show novel views of the predicted 3D hand pose.

5. Implementation details

Training details: We train the networks Φ, Ω, Λ and D, from scratch according to the loss function (Equation 12) from the main paper. We use the Adam optimiser [6] with learning rate of 2×10^{-4}, and $\beta_1 = 0.5, \beta_2 = 0.999$. Each batch is formed by sampling from the images and randomly sampling from the prior of unpaired 2D poses (which is then transformed to a skeleton image). The batch size is 96. Our model was trained for around 40 hours using one GPU from a NVIDIA DGX-MAX-Q server. The NF is pre-trained in line with [11] as shown on the next section.

Model components: This section shows the details of the networks used in our model. We include a pictorial representation of all the networks shown in Figure 2 from the main paper. The upper part of Figure 7 displays the networks needed for the mapping from image x to 3D pose v. The lower part shows the discriminator D needed during training to evaluate the skeleton images. In particular, Φ and Ω are based on [4, 5], the discriminator D on [5, 12], and the lifting network Λ on [8, 11].

Following [5], with respect to the discriminator D, we use three identical convolutional architectures, inputting different scales of the image: the original image and its downsized versions by $\frac{1}{2}$ and $\frac{1}{4}$, respectively. We take the mean of the patchwise outputs from the three network, as indicated in Figure 7. The normalising flow network is shown in the following subsection since it requires a more detailed explanation.

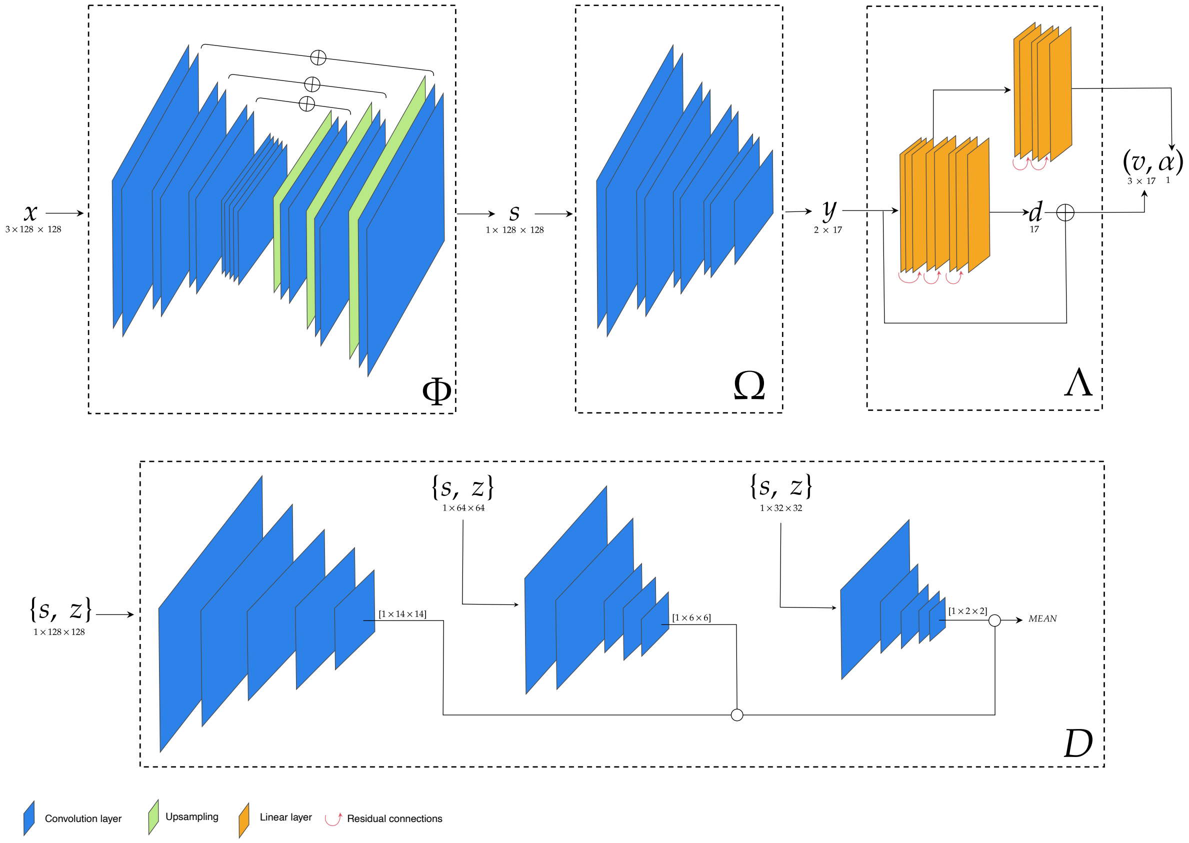

Figure 7. **Pictorial representation of the networks that integrate our model.** Blue rectangles represent convolutional layers, and the orange ones the linear layers. Note that to keep the diagrams as simple as possible, we omit some components, such as the size of layers, normalisation layers, and activation functions; we include these elements in Table 2, Table 3, Table 4, and Table 5 of the next section.

Implementation details

Normalising flow: Following [11], we use the network in [2] to represent $f(\bar{y})$ from Equation 7 in the main paper. This network consists of consecutive affine coupling blocks like the one shown in Figure 8. Each coupling block applies a random permutation of the input. In our case, the input $\bar{y}$ is the image in the PCA subspace of the 2D pose $\hat{y}$. After the permutation, it splits the vector into two parts, m_1 and m_2. The first part m_1, is used to predict a scale s and a translation t to deform m_2. In the end, w_1 (or m_1 since it remains unchanged) is concatenated with the deformed m_2 represented as w_2.

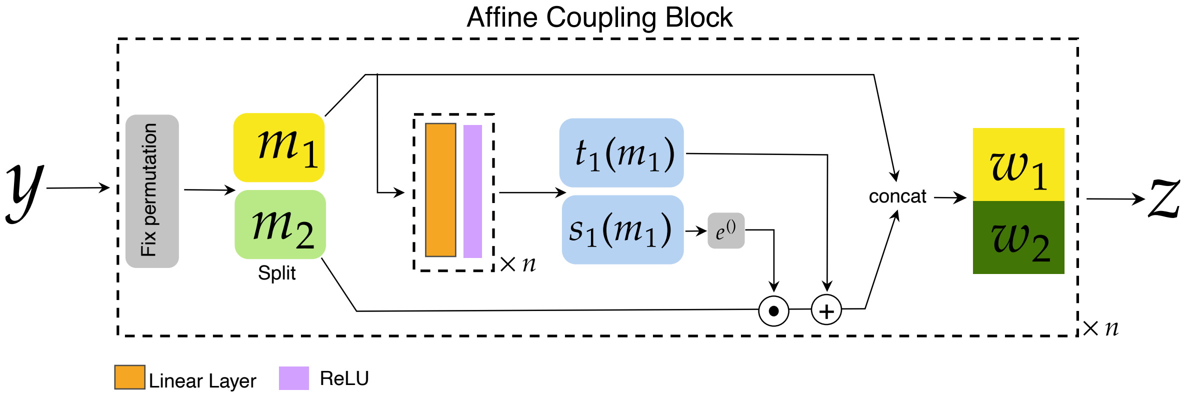

Figure 8. **Affine coupling block.** Multiple consecutive coupling blocks integrates the normalising flow. Diagram adapted from the supplemental material of [11].

During the forward pass the scale s and translation t are calculated as a function of m_1, and then used to deform m_2 as follow

$$w_2 = \exp(s(m_1))m_2 + t(m_1) \qquad \& \qquad w_1 = m_1 \tag{1}$$

Similarly, the backward part is defined by

$$m_1 = w_1 \qquad \& \qquad m_2 = (w_2 - t(w_1))\exp(-s(w_1)) \tag{2}$$

The determinant of the Jacobian is given by

$$\det\left(\frac{\partial f}{\partial \bar{y}}\right) = \exp\left(\sum_j s(m_1)_j\right). \tag{3}$$

Since the Jacobian of f does not need to calculate the Jacobian of the scale s and translation t functions, these could be complex.

Implementation details

Networks Structure

Layer	Out.Shape	Act.	Norm.
Conv2d	$32 \times 128 \times 128$	ReLU	Batch
Conv2d	$32 \times 128 \times 128$	ReLU	Batch
Conv2d	$64 \times 64 \times 64$	ReLU	Batch
Conv2d	$64 \times 64 \times 64$	ReLU	Batch
Conv2d	$128 \times 32 \times 32$	ReLU	Batch
Conv2d	$128 \times 32 \times 32$	ReLU	Batch
Conv2d	$256 \times 16 \times 16$	ReLU	Batch
Conv2d	$256 \times 16 \times 16$	ReLU	Batch
Conv2d	$256 \times 16 \times 16$	-	-
Conv2d	$256 \times 16 \times 16$	ReLU	Batch
Conv2d	$256 \times 16 \times 16$	ReLU	Batch
Upsampling	$128 \times 32 \times 32$	-	-
Conv2d	$128 \times 32 \times 32$	ReLU	Batch
Conv2d	$128 \times 32 \times 32$	ReLU	Batch
Upsampling	$64 \times 64 \times 64$	-	-
Conv2d	$64 \times 64 \times 64$	ReLU	Batch
Conv2d	$64 \times 64 \times 64$	ReLU	Batch
Upsampling	$32 \times 128 \times 128$	-	-
Conv2d	$32 \times 128 \times 128$	ReLU	Batch
Conv2d	$1 \times 128 \times 128$	-	-
Final output shape: $1 \times 128 \times 128$			

Table 2. **Structure of network Φ.**

Layer	Out.Shape	Params	Act.	Norm.
Conv2d	$32 \times 128 \times 128$	1,600	ReLU	Inst.
Conv2d	$32 \times 128 \times 128$	9,248	ReLU	Inst.
Conv2d	$64 \times 64 \times 64$	18,496	ReLU	Inst.
Conv2d	$64 \times 64 \times 64$	36,928	ReLU	Inst.
Conv2d	$128 \times 32 \times 32$	73,856	ReLU	Inst.
Conv2d	$128 \times 32 \times 32$	147,584	ReLU	Inst.
Conv2d	$256 \times 16 \times 16$	295,168	ReLU	Inst.
Conv2d	$256 \times 16 \times 16$	590,080	ReLU	Inst.
Conv2d	$17 \times 16 \times 16$	4,369	None	None
Final output shape: $17 \times 16 \times 16$				
Total params: 1,177,329				

Table 3. **Structure of network Ω.**

Layer	Out.Shape	Params	Act.	Norm.
Linear	1×1024	35,840	LReLU	None
Linear	1×1024	1,049,600	LReLU	None
Linear	1×1024	1,049,600	LReLU	None
Linear	1×1024	1,049,600	LReLU	None
Linear	1×1024	1,049,600	LReLU	None
Linear	1×1024	1,049,600	LReLU	None
Linear	1×1024	1,049,600	LReLU	None
Linear	1×17	17,425	LReLU	None
Linear	1×1024	1,049,600	LReLU	None
Linear	1×1024	1,049,600	yReLU	None
Linear	1×1024	1,049,600	LReLU	None
Linear	1×1024	1,049,600	LReLU	None
Linear	1×1	1,025	LReLU	None
Final output shape: $[[1 \times 17], [1 \times 1]]$				
Total params: 10,550,290				

Table 4. **Structure of network Λ.**

Layer	Out.Shape	Params	Act.	Norm.
Conv2d	$64 \times 64 \times 64$	1,088	LReLU	None
Conv2d	$128 \times 32 \times 32$	131,200	LReLU	Inst.
Conv2d	$256 \times 16 \times 16$	524,544	LReLU	Inst.
Conv2d	$512 \times 15 \times 15$	2,097,664	LReLU	Inst.
Conv2d	$1 \times 14 \times 14$	8,193	None	None
Conv2d	$64 \times 32 \times 32$	1,088	LReLU	None
Conv2d	$128 \times 16 \times 16$	131,200	LReLU	Inst.
Conv2d	$256 \times 8 \times 8$	524,544	LReLU	Inst.
Conv2d	$512 \times 7 \times 7$	2,097,664	LReLU	Inst.
Conv2d	$1 \times 6 \times 6$	8,193	None	None
Conv2d	$64 \times 16 \times 16$	1,088	LReLU	None
Conv2d	$128 \times 8 \times 8$	131,200	LReLU	Inst.
Conv2d	$256 \times 4 \times 4$	524,544	LReLU	Inst.
Conv2d	$512 \times 3 \times 3$	2,097,664	LReLU	Inst.
Conv2d	$1 \times 2 \times 2$	8,193	None	None
Final output shape: $[[1 \times 14 \times 14], [1 \times 6 \times 6], [1 \times 2 \times 2]]$				
Total params: 8,288,067				

Table 5. **Structure of network D.**

References

[1] Ching-Hang Chen and Deva Ramanan. 3d human pose estimation= 2d pose estimation+ matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7035–7043, 2017. 1

[2] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016. 8

[3] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, jul 2014. 1

[4] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Unsupervised learning of object landmarks through conditional image generation. *Advances in neural information processing systems*, 31, 2018. 7

[5] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Self-supervised learning of interpretable keypoints from unlabelled videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8787–8797, 2020. 7

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 7

[7] Jogendra Nath Kundu, Siddharth Seth, MV Rahul, Mugalodi Rakesh, Venkatesh Babu Radhakrishnan, and Anirban Chakraborty. Kinematic-structure-preserved representation for unsupervised 3d human pose estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 11312–11319, 2020. 1

[8] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 2640–2649, 2017. 7

[9] Dushyant Mehta, Helge Rhodin, Dan Casas, Pascal Fua, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt. Monocular 3d human pose estimation in the wild using improved cnn supervision. In *2017 international conference on 3D vision (3DV)*, pages 506–516. IEEE, 2017. 1

[10] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1145–1153, 2017. 1

[11] Bastian Wandt, James J Little, and Helge Rhodin. Elepose: Unsupervised 3d human pose estimation by predicting camera elevation and learning normalizing flows on 2d poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6635–6645, 2022. 7, 8

[12] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 7