Occlusion-tolerant and personalized 3D human pose estimation in RGB images

Ammar Qammaz
Computer Science Department, Univ. of Crete, and
Institute of Computer Science, FORTH, Greece
Email: ammarkov@ics.forth.gr

Antonis Argyros
Computer Science Department, Univ. of Crete, and
Institute of Computer Science, FORTH, Greece
Email: argyros@ics.forth.gr

***Abstract*—We introduce a real-time method that estimates the 3D human pose directly in the popular Bio Vision Hierarchy (BVH) format, given estimations of the 2D body joints originating from monocular color images. Our contributions include: (a) A novel and compact 2D pose representation. (b) A human body orientation classifier and an ensemble of orientation-tuned neural networks that regress the 3D human pose by also allowing for the decomposition of the body to an upper and lower kinematic hierarchy. This permits the recovery of the human pose even in the case of significant occlusions. (c) An efficient Inverse Kinematics solver that refines the neural-network-based solution providing 3D human pose estimations that are consistent with the limb sizes of a target person (if known). All the above yield a 33% accuracy improvement on the Human 3.6 Million (H3.6M) dataset compared to the baseline method (MocapNET) while maintaining real-time performance (70 fps in CPU-only execution).**

I. Introduction

Human body pose estimation/recovery is a very active research topic with a great variety of important applications. Recent innovations using deep learning-based approaches [1] have demonstrated remarkable results. However, despite the significant improvements in accuracy and performance, they have not yet penetrated the commercial motion capture (MOCAP) market that still relies on systems that use cumbersome MOCAP suits with inertial measuring units (IMUs) [2] or physical markers and expensive multi-camera setups [3].

In this work we are particularly interested in real-time methods for markerless human motion capture from RGB images. We present a method that is inspired by the MocapNET 3D pose estimator [7], the first end-to-end neural network to directly convert 2D point clouds to BVH motion frames. The improvements over MOCAPNET are threefold. First, we propose a neural network ensemble that uses a novel 2D pose representation we have coined Normalized Signed Rotation Matrices (NSRMs). NSRMs require 16% less network parameters compared to the baseline method. Second, the proposed network is much more robust to occlusions, as it allows the decoupling of the human body into two separate (upper, lower) kinematic hierarchies. Finally, we have developed a novel and efficient Inverse Kinematics (IK) solver that refines the DNN-based solution by taking into account possibly known limb dimensions and camera parameters. Extensive experimental results in standard datasets demonstrate that the resulting method achieves a 33% accuracy improvement over the baseline method, while maintaining real-time execution at 70 fps in CPU only execution. Moreover, extensive qualitative experiments (see Figure 1) show that our work manages to capture accurately the 3D pose of humans in RGB data acquired “in the wild”.

II. Related Work

Advancements in neural networks for image classification [8], [9] inspired research on human pose estimation tasks. Early notable works like DeepPose [10] led to more mature works like OpenPose [11], [12] that was the first method to robustly and accurately solve the “in-the-wild” 2D pose estimation problem in real-time. As more methods effectively dealt with the 2D pose estimation problem, the research frontier naturally progressed towards 3D pose estimation. Several methods attempt 3D pose estimation in one step, operating directly on RGB images. An excellent recent survey of deep-learning methods for monocular pose estimation from RGB sources is provided by Chen et al. [1]. 3D human pose estimation methods include interesting recent works like LCR-Net [13], [14], DensePose [15] and a variety of other methods [16], [17], [18], [19], [20], [21] that achieved impressive 3D pose estimation results directly from RGB data.

Our work belongs to the so-called “two-stage” approaches. The first stage of these methods extracts 2D human joints which are then lifted in 3D in the second stage. The method we present in this paper is inspired by MocapNET [7] which we consider as our baseline due to its very fast estimation speeds and its direct BVH output. Another similar work is [22] that utilizes the Euclidean Distance Matrix (EDM) encoding as its input representation. Other recent relevant works include [23] which directly models joint connections instead of having an exhaustive joint-to-joint relation map to offer more features for the neural network. Translation and rotation invariant features have also been suggested [24], as well as representations that handle rotation discontinuities [25] and works that better encode structural properties such as kinematic chain spaces [26]. The neural network we use shares the compact formulation of [27] although it does not use residual connections or physics simulations [28]. Although our method uses a BVH armature [29] that targets the motionbuilder [30] armature, many two-stage methods [31], [32] utilize the SMPL [33] linear model and regress both 3D pose as well as 3D shape.

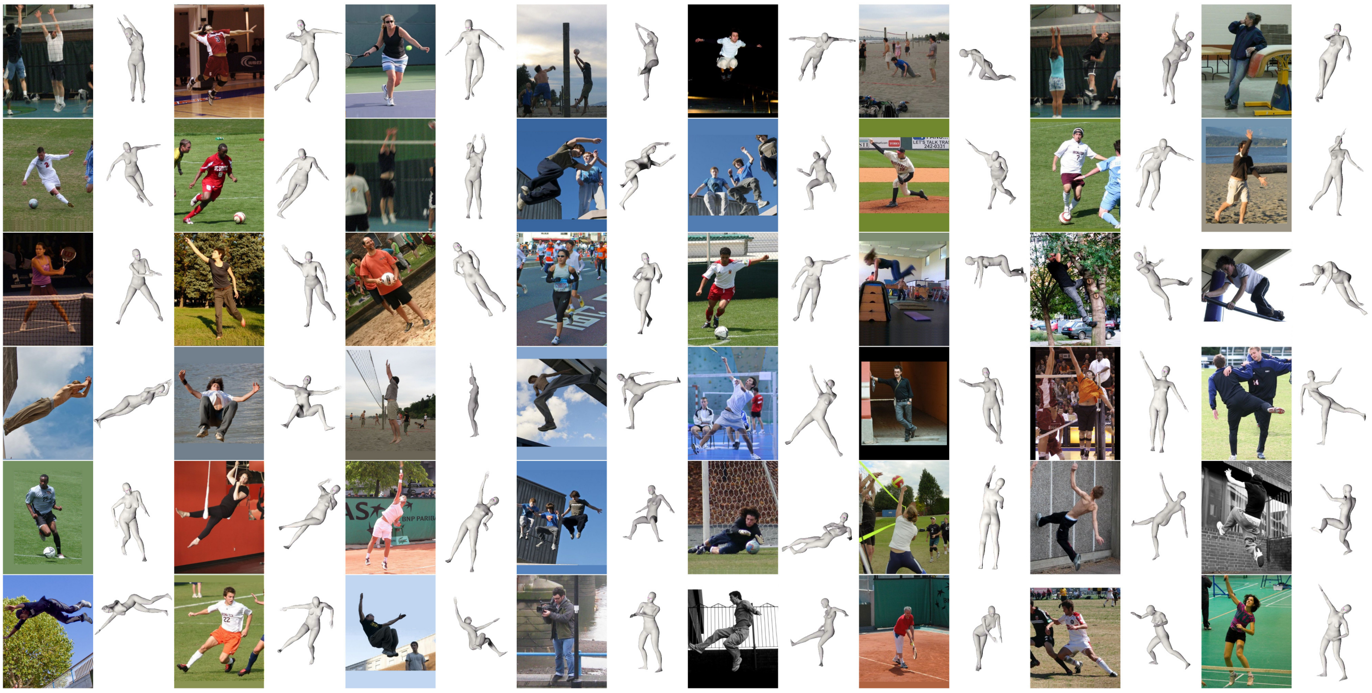

Fig. 1. Qualitative results when testing our method on the Leeds sport dataset [4]. Our BVH output is rendered using Blender [5] and MakeHuman [6].

Other methods [34], [35] use different models and perform regression on monocular data that, however, originate from an RGBD camera. Recent trends show the importance of high-level inference like the one we attempt since state of the art RGBD methods now even account for garments [36] and RGB methods have been proposed that combine body, hand and face pose estimation [37].

The Hierarchical Coordinate Descent (HCD) Inverse Kinematics (IK) module we present shares some similarities with FABRIK [38], however we only perform forward steps and we do not use 3D conic sections as a heuristic to improve 3D angles. Our method is inspired by non-linear least squared methods that are used by methods like CERES [39] and their internal implementation of the Levenberg-Marquardt [40], [41] algorithm. It is also conceptually similar to the semi-stochastic coordinate descent [42], however we do not rely on random selection of joints but instead use a hierarchical approach that processes all joints.

III. Methodology

Our method (see Figure 2) achieves 3D human pose estimation in three discrete steps, (a) computation of 2D joints and encoding them using a Normalized Signed Rotation Matrice (NSRM), (b) classification of a NSRM-based 2D human pose into four orientation classes and use of appropriate neural networks to regress it to a 3D pose and (c) Inverse Kinematics for fine-tuning and personalization of the obtained results. The neural networks in step (b) need to be trained in appropriate data sets containing ground truth by also employing data augmentation techniques. The output of our method is a BVH [29] file with 498 motion fields. This can be used to animate and render any rigged 3D mesh since this representation is compatible with a wide variety of 3D editing applications like Blender [5] and popular 3D graphics engines like Unity, etc. The BVH motion fields correspond to the degrees of freedom of the depicted human armature and can accommodate pose data about the body, the human face, hands as well as feet. The proposed body pose estimation method currently only populates 87 of the degrees of freedom of the armature, leaving the rest to their default values. The basic components of the above steps are described below in more detail.

Estimation of 2D joints: We use the OpenPose pose estimator [11], [12] on incoming RGB images to produce 2D human joints in the popular BODY25 [12], [43] format.

Normalized Signed Rotation Matrices (NSRMs): The estimated 2D joints hierarchy is encoded into two Normalized Signed Rotation Matrices (NSRMs), one for the upper body and one for the lower body. An NSRM is a novel scheme to encode 2D poses that is translation and scale invariant and efficient in terms of the total element count required to encode a 2D pose. It is conceptually similar to Euclidean Distance Matrices (EDMs) [44] and Normalized Signed Distance Matrices (NSDMs) [7]. EDMs are simple data representations that encode joints in relation to each other by storing their Euclidean distances. This makes poses invariant to translation. NSDMs are normalized to be invariant also to scale changes. However, their disadvantage is that they require two encoding channels, one for X and one for Y joint coordinates. NSRMs share the same design considerations as the other encodings while offering a lower total parameter count.

To describe NSRMs we must first further describe our input.

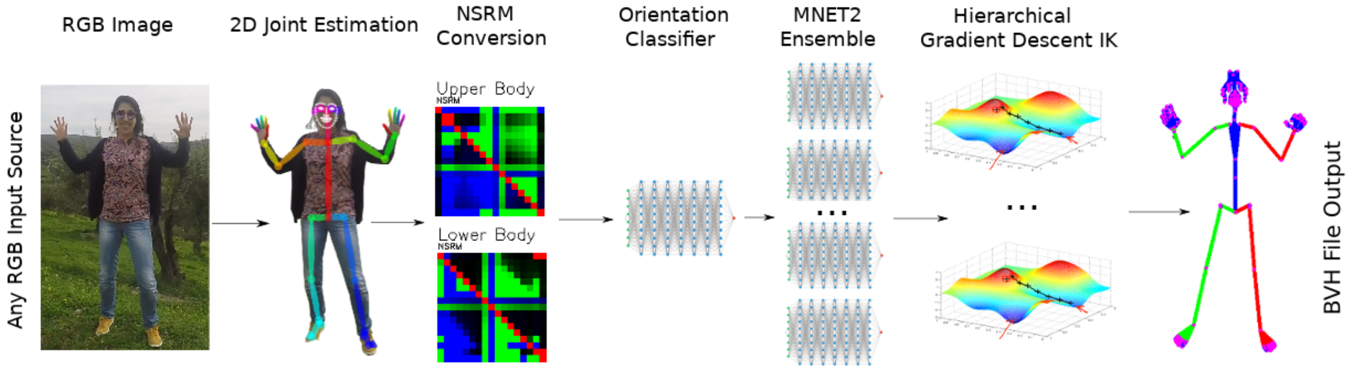

Fig. 2. Overview of the proposed method. A 2D human pose extracted from an RGB image using OpenPose [12] is converted to two Normalized Signed Rotation Matrices (NSRMs) encoding the upper and lower body. A classifier uses these NSRMs to identify the orientation of the observed person. We choose an ensemble specifically trained for the classified orientation to convert the NSRMs to a BVH pose. Finally, we refine the BVH pose by inverse kinematics (IK) with optionally known limb dimensions and camera configuration.

The employed BODY25 [43] 2D joint input consists of 25 2D points $J_{2D} = \{p_1, ..., p_{25}\}$ [12]. Out of those, we select subsets of 2D joints to create an NSRM matrix. In particular, the body hierarchy is split in upper and lower body to combat occlusions, so we define one NSRM for each part. To derive the upper body $NSRM^u$ we use joints hip, l/r eye, neck, l/r shoulder, l/r elbow and l/r hand. For the lower body $NSRM^l$ joints used are hip, r/l thigh, r/l knee, r/l heel and r/l big toe.

An NSRM associated with M joints is constructed as follows. The coordinates (a_x, a_y) of a participating 2D joint a are normalized to the input image frame dimensions and are thus bounded in the range $[0, 1]$. We also associate each such joint with a visibility parameter a_v provided by thresholding the OpenPose joint confidence values (1 if joint is visible, 0 if joint is occluded).

For each pair of 2D points a, b we can declare a new point $c = (b_x, b_y - \mid b - a \mid)$ that is the point b translated vertically by the length of vector ab. Using these three points and the atan2 function [45] we can encode (Equation 1) the relation between points a and b as well as their relative rotation towards a fixed vertical axis, that is:

$$NSRM^h(a,b,c) = \begin{cases} atan2(A_xB_y - A_yB_x, A_xB_x + A_yB_y) \\ a \neq b, \\ \\ 0, otherwise, \end{cases} \tag{1}$$

where $A_x = b_x - a_x$, $A_y = b_y - a_y$, $B_x = c_x - b_x$ and $B_y = c_y - b_y$. $NSRM^h(a,b,c)$ is invariant to skeleton translation and scale. The representation encodes the relative position of joints (albeit using the rotation formed from triangle $a\hat{b}c$), as well as orientation (since bc is parallel to the y axis of the world). Finally, joint order is preserved through the sign of the *atan2* function. An advantage of this encoding compared to NSDMs [7] is that we can easily force alignment of all retrieved angles using a pivot point and rotation. In our body pose estimation scenario where humans typically stand upright this is not a very important characteristic but we predict that hand pose estimation using NSRMs might benefit from the possibility of controlling the encoded rotation of the 2D joints by always aligning input matrices to a pivot point (e.g. hip to neck) in a way that makes the descriptor rotation invariant.

$NSRM^u$ for upper and $NSRM^l$ for lower body are formed by computing the respective NSRM for all joint pairs of the corresponding body part. Having all pairwise joint relations encoded in a matrix means that available features for the neural network can be readily leveraged in a relatively shallow and thin network without requiring too many operations. A great improvement compared to NSDMs [7] is that an NSRM is encoded in a single channel, as opposed to NSDMs that require two, one for the x and one for y joint coordinates. Thus, NSRMs use half the element count compared to NSDMs. For dense neural networks this amounts to a drastic parameter decrease.

The neural network ensemble: The NSRMs encoding a human skeleton are given as input to an orientation classifier that decides on the orientation of the depicted person. Depending on classifier output, we select the most qualified neural network ensemble to regress the NSRM matrices to a 3D pose. For our novel ensemble orientation classifier we use an 8 hidden layer, densely connected network with 322, 161, 107, 80, 64, 46, 40 and finally 36 parameters in each layer and SeLU activations starting from a dropout of 20% and set to 40% dropout after the 4th layer. The final layer uses a softmax activation to produce compatible results for our categorical cross-entropy loss function. Orientations are encoded using a one-hot encoding with four categories (front, back, left and right) with the correct category set to 1 and the rest set to 0 for our training samples. While using our classifier we use a winner-takes-all strategy treating the highest scoring classification as the correct orientation.

The neural network for 3D pose estimation uses 6 hidden layer self normalizing neural networks (SNNs [46]) of 152K weights for each encoder, $\sim 6.3M$ parameters for each joint hierarchy, amounting to ~$12.7M$ parameters for the aggregate ensemble as seen in Figure 3. This is a ~16% improvement in

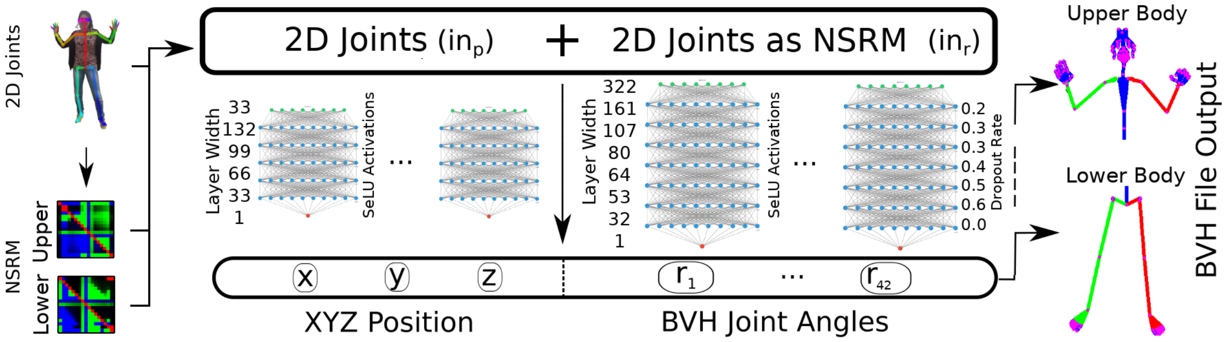

Fig. 3. Overview of a neural network ensemble trained for a particular orientation class. We use 4 hidden layers for the SNNs [46] that regress the 3D skeleton position and 7 layer SNN encoders to retrieve 3D angles. Layer widths and dropout rates are listed for each layer. Values 'in_p = 33, in_r = 289 refer to the number of layer input elements.

parameter count over the baseline network that had ensembles of 270K weights and a ~15M parameter network.

An important design characteristic of our proposed network is that the upper body hierarchy is completely independent from the lower body part. This means that even in cases of extreme occlusions (i.e., having the whole lower body occluded) our method can still produce accurate pose estimation results for the upper body. This is in contrast to the baseline method that would completely fail in this case.

Leveraging the findings of research on neural networks [47], [48] we have introduced a much higher degree of dropout that starts at 20% and reaches 40% as seen in Figure 3 instead of the flat 20% dropout rate of the baseline method [7]. This seems to also have a positive effect on occlusion handling since the trained network learns to derive accurate results even if a big part of the data is missing. This also seems to amplify the benefits of the occlusion resistant properties of the separation of upper and lower body.

Besides SNNs, we performed experiments with many other candidate neural networks. Particularly interesting were experiments using convolutional layers that allowed similar training accuracy to the baseline dense network using only 15K weights for each encoder. However, such convolutional encoders proved to be prone to overfitting and produced noisy results after thorough testing on validation data.

Training the ensemble: We use Keras [49] and Tensorflow [50] as our deep-learning framework. Our networks are trained using the RMSProp optimizer with a batch size of 128, learning rate of 0.0002, $e = 10^{-6}$ and employing a variable epoch configuration depending on the difficulty of the joint. Hips and shoulders are considered difficult joints and are trained for 20 epochs. Elbows, knees, chest and neck joints are considered medium difficulty joints and are trained for 15 epochs, while the rest of the joints are considered easy and trained for 10 epochs. The loss function we use is mean squared error (MSE) between predicted and ground truth 3D joint rotations. We train one encoder for each degree of freedom of each joint hierarchy for each orientation class. SNN [46] layers are initialized with random samples from a truncated normal distribution centered at 0 with $\sigma = \sqrt{1/N}$ where N is the number of input units in the weight tensor. As we gradually train encoders, instead of starting from scratch, we load the weights of neighboring joint encoders to retain knowledge previously acquired in our training session. We perform early-stopping by monitoring loss and terminating training if loss delta is less than 0.001 in 5 or more consecutive training epochs. We also use model checkpoints [51] so that each training session returns the best loss achieved, regardless of the epoch it was encountered. This helps against overfitting.

Training dataset filtering: To train the neural network part of our method we employ the BVH conversion [52] of the Carnegie Mellon University MOCAP dataset [53]. We use datasets 1 to 144 each of which contain various actions like jumping, dancing, walking and climbing. A pitfall we encountered is that 187 of the 2535 BVH files [52] have corrupted arm or leg angles, so we manually discarded incorrect files[1]. We also appended the original joint hierarchy with a complete facial rig as well as feet that also model toes.

The BVH files have some degree of repetition because they are recorded in high frame rate and because each action is repeated several times (see Figure 4). This skews the training procedure as it overemphasizes over-represented poses against some interesting, under-represented ones. Thus, we filtered the original dataset (see Figure 4) and discarded 30% of the poses where all joints where clustered around the same positions, leaving approximately 2.2M training poses. This is another improvement compared to the baseline [7] method that, due to no dataset pose filtering, had to be trained on a much smaller selection of CMU actions leading to poorer overall training pose diversity. The clustering tool is available in the github repository [54].

Training dataset augmentation: The dataset depicts persons performing actions in fixed trajectories, so to further enrich it we augment it by randomizing the location of the observed skeleton for each frame. Directly randomizing the 3D coordinates of the skeletons leads to sub-optimal 2D coverage of the

[1] The modified dataset including the flagged incorrect pose files described will become publicly available for download.

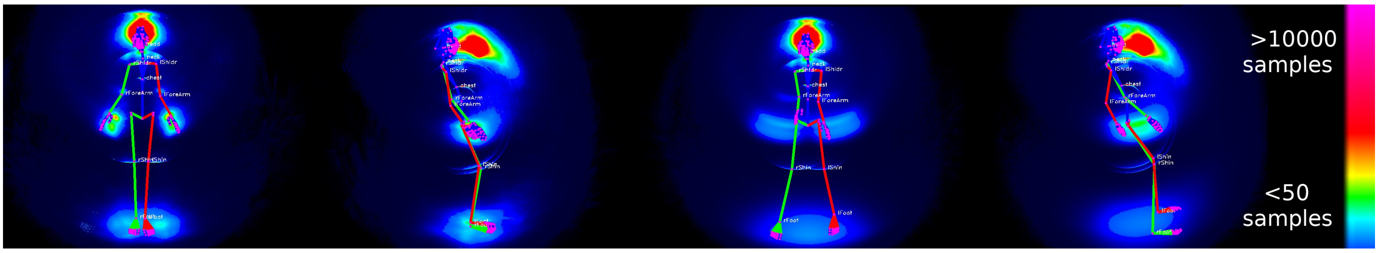

Fig. 4. Joint location heatmaps of the 3.9M poses of the CMU dataset after translation and rotation normalization. Left: frontal and side illustration of the accumulated joint frequencies in the raw dataset. Right: same information, after dataset filtering and augmentation.

input frame. Instead, we pick a random 2D point on the virtual camera frame and then pick a random depth value to create our randomized point. This way the randomization covers more uniformly the whole view frustrum.

A second data augmentation procedure diversifies the recorded 3D joint configurations. The perturbations use uniform random values so that the new value is at most $\pm x°$ away from the original orientation. We perturb the r/l shoulder by ±30°, r/l elbow ±16°, abdomen and chest by ±10°, r/l hip ±30° and r/l knee ±10°.

A final data augmentation concerns the orientation randomization of the human skeletons on the basis of the four considered orientation classes. Therefore, we split randomization in four quadrants (front, back, left, right). All quadrants have the same limits for rotations on the x and z axis ($-35° \leq r_x \leq 35°$ and $-35° \leq r_z \leq 35°$). The orientation r_y is split into overlapping quadrants of 100° each, to ensure proper handling in the case of marginally inexact orientation classification. This class separation scheme not only allows smaller, more accurate and higher-performance neural networks that have an easier task to accomplish, but also mitigates neural network learning problems due to angle discontinuities [25].

These filtering and augmentation processes result in $\sim 2.2M$ poses per orientation class. Thus, the final "ensemble of ensembles" is trained with over $8.8M$ poses, however, without needing to generalize to all of them at once or contain them all in RAM during training.

Hierarchical Coordinate Descent (HCD) inverse kinematics: The inverse kinematics solver is used to refine the pose regressed by the neural network. IK solvers typically rely on a non-linear least squares optimizer, with the CERES [39] solver being a very popular choice. However, from a computational point of view, this is an expensive operation. In our 3D human pose estimation problem, we obtain consistently, fairly accurate joint estimations. Moreover, each neural network encoder is conditionally independent from the others. This suggests the appropriateness of an iterative solution to the pose refinement problem. Additional inspiration comes from the performance of efficient, heuristic IK methods like FABRIK [38] that iteratively traverse the kinematic chain by making individual improvements to each joint. Specifically, we think of our IK problem as the refinement of a hypothesis vector h that consists of individual 3D human pose parameters resulting from the neural network of the previous step. We also consider the objective function E_{2DJ} that quantifies the mean squared error (MSE) of the 2D joints projected, compared to the 2D joints observed.

$$E_{J2D}(h, o) = \frac{1}{m_J} \sum_{i=1}^{m_J} |j_i^h - j_i^o|^2. \tag{2}$$

We assume that changes in, e.g., the parameters of the left arm will not affect errors on the right leg, therefore we decompose the human body into 6 kinematic chains. The first kinematic chain C_1 consists of hips, shoulders and neck. The rest of the kinematic chains are C_2 (abdomen, neck, shoulders), C_3 (right shoulder, elbow, hand), C_4 (left shoulder, elbow, hand), C_5 (right hip, knee, heel, toe) and C_6 (left hip, knee, heel, toe).

For a certain kinematic chain, we define an iterative error minimization scheme. At the n^{th} iteration of this process, we modify each parameter c of the chain by d_c^n defined as:

$$d_c^n = \beta d_c^{n-1} + l_r \left(\frac{E_{J2D}(h_{n-1}, o) - E_{J2D}(h_n, o)}{2(d_c^{n-1} + e)} \right). \tag{3}$$

In the above equation, β is a momentum control parameter we set to 0.9 and $e = 0.0001$ is used to avoid division by zero. $l_r = 0.001$ is a learning-rate-type of parameter that controls the rate at which the change of the error affects the change of a parameter. For a certain joint of the kinematic chain, we alternate between its x, y and z rotational parameters for 30 epochs and only accept combined value updates if the achieved objective function is improved compared to the initial starting point. We finish the procedure after going through every joint of the kinematic chain for 5 iterations. We experimentally identified the 5 iteration sweet-spot after synthetic experiments on CMU [53] data as seen in Fig 5 (right). Kinematic chains are optimized in groups, with the first being C_1, C_2, followed by C_3 to C_6 which can be considered in parallel.

IV. Experiments

We base the quantitative evaluation of the proposed method on the Human 3.6M (H36M) [55] dataset which is used to compare a variety of methods [37], [7], [66], [67], [58], [16], [59], [60], [61], [62], [63], [19], [13], [64], [35], [17], [68]. Evaluation on H36M is performed through specified protocols that use mean per joint estimation error (MPJPE) after Procrustes alignment [69] of the output of a method compared to the ground truth. The H36M protocol 1 dictates training on subjects 1, 4, 6, 7, 8 and testing on subjects 9 and 11 on 2D points originating from all available cameras.

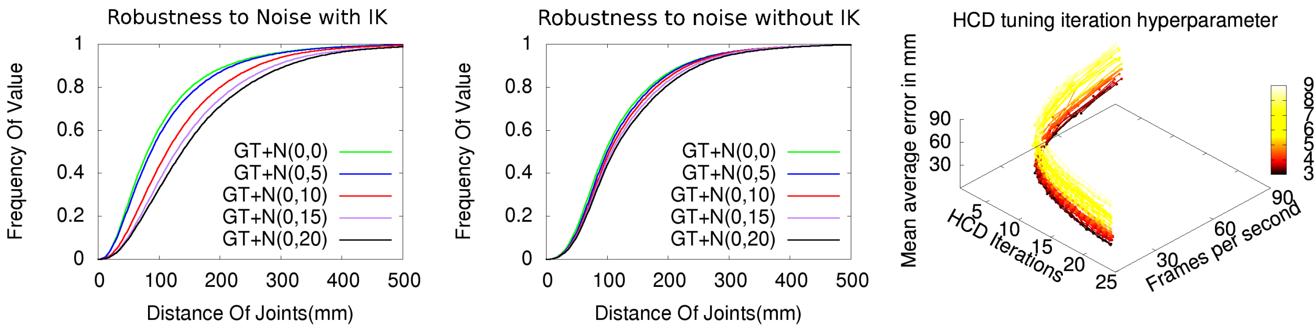

Fig. 5. Proposed method accuracy **with** (left) and **without** (middle) the HCD IK module for various levels of Gaussian noise on H36M [55] 2D input. **Right:** Synthetic experiments on CMU [53] data. Varying the HCD iterations parameter reveals a performance/accuracy sweet-spot at 5 iterations.

Input	Dir	Dis	Eat	Gre	Pho	Pos	Pur	Sit	Smo	Pho	Wai	Wal	Dog	WaT	Sit.	Avg
Ours (NN+HCD)	69	78	92	78	100	79	134	141	97	89	84	85	102	81	165	**108**
Ours (NN only)	88	105	116	99	120	102	152	165	127	116	114	112	146	98	180	**122**
MocapNET [7]	135	140	145	143	153	137	174	215	156	150	151	156	166	134	246	**160**

TABLE I
COMPARISON OF OUR METHOD WITH THE BASELINE APPROACH [7] WITH RESPECT TO THE MPJPE ERROR METRIC. METHODS ARE TRAINED ON CMU AND TESTED USING H36M BLIND PROTOCOL 1.

[55]	[7]	**NN**	[56]	[57]	[58]	[16]	[59]	**NN+HCD**	[60]	[61]	[62]	[63]	[19]	[13]	[64]	[35]	[17]	[65]
162	160	122	119	118	116	113	108	**108**	107	101	93	88	88	88	82	80	72	40
N/A	2.5	2.0	N/A	<0.1	<0.1	<0.1	<0.1	**0.6**	N/A	N/A	0.01	1.8	0.28	N/A	0.46	0.38	N/A	N/A

TABLE II
COMPARISON OF METHODS TESTED ON H36M PROTOCOL 1. 1ST ROW: MPJPE IN MM (THE SMALLER, THE BETTER), 2ND ROW: RATIO OF ACHIEVED FRAME RATE OVER MPJPE (THE LARGER, THE BETTER).

An important consideration is that the baseline method we are improving upon [7] does not train on any subjects provided by H36M. Therefore, in order to assure a fair comparison, we also performed experiments without training on H36M samples. Hence, following [7], we also label the protocol for our quantitative experiments as *Blind P1 (BP1)*.

Comparison to the baseline: The obtained results are summarized in Table I. The compared methods are (a) *NN+HCD*, the full proposed solution, (b) *NN*, the estimation provided by the neural network component of our method (no HCD) and (c) *MocapNET*, the baseline approach [7]. Table I reveals that the proposed solution (*NN+HCD*) is superior compared both to the network-only solution and to the baseline, by a great margin. Importantly, the network-only solution of our approach, is also superior to the baseline, a fact that we attribute to our superior 2D joints encoding, the twice as many orientation categories and the more elaborate orientation classifier. H3.6M [55] quantitative tests do not feature scene occluders so occlusions are tested using in-the-wild videos with cases of severe occlusions (last 3 examples on bottom row of Fig 6) the baseline method predictably breaks down providing incoherent results, since its single NSDM [7] matrices are architecturaly not designed to deal with this scenario. Each missing joint eliminates one line and one column of the matrix and even with a few occlusions matrices become extremely sparse causing neural network convolutions to only produce corrupted poses. The proposed approach however is much more robust to these scenarios.

Evaluation with respect to noise tolerance: In order to assess the robustness of our approach to noise, we repeated the above evaluation assuming different levels of noise contamination of the input 2D joints. Specifically, we considered ground truth 2D joint positions contaminated errors following the normal distribution $\mathcal{N}(\mu, \sigma^2)$. Figure 5 illustrates the relevant results by showing the percentage of joints that are estimated within a certain distance from their ground truth positions for different noise levels. We observe that when 2D joints are accurate, the inverse kinematics provides a consistent improvement. As noise becomes more intense, the performance are degraded due to the effort of the Inverse Kinematics module to strictly comply to the corrupted input data. This is in contrast to the neural network which responds in a more timid way performing some kind of internal pose filtering.

Comparison to SoTA: Comparison to other methods is summarized in Table II. Keeping in mind that the neural network ensemble is evaluated at sustained rates of over 250fps on CPU execution when executed on a relatively dated i7-4790 CPU and over 70fps when using both the neural network and the unoptimized HCD module, we believe that we achieve a very good balance in the performance/accuracy trade-off. Moreover, we stress that, contrary to the rest of the evaluated

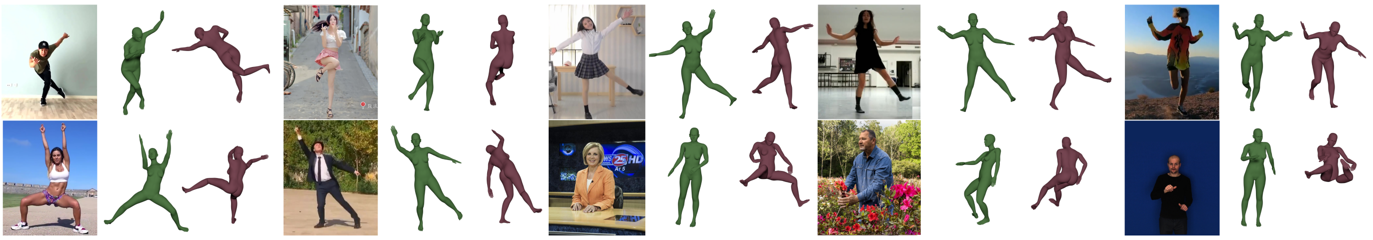

Fig. 6. Qualitative results of our method (green) compared to the baseline [7] (red) when tested on "in-the-wild" YouTube videos. We observe improved accuracy, robust orientation classification and better occlusion tolerance.

methods, our method has not been trained with *any* of the H3.6M data, therefore, the comparison is disadvantageous to our method. Last but not least, our method always outputs anatomically valid results that comply to the same joint dimensions, compared to methods that ignore joint dimensions and relevant constraints.

Qualitative evaluation: For the qualitative assessment of the proposed method we used RGB videos collected from the web. The BVH files that were output by our method were loaded in Blender [5] where we animated a skinned model created using MakeHuman [6] to visualize the results. Indicative results of single frame pose estimation can be seen in Figure 1. Still shots from the aforementioned YouTube videos can be seen in Figure 6 in comparison to the results of the baseline approach [7]. We observe improved accuracy over the baseline approach, especially in the case of considerable joint occlusions. Failure cases arise when supplying erroneous focal lengths to the Inverse Kinematics module which then tends to make persons bend forward in order to satisfy the 2D joint constraints while also adhering to the supplied focal lengths. Further results are provided in the supplementary material accompanying the paper[2].

V. Discussion

We presented a series of novel ideas and methods that allow 3D human pose estimation at a 33% accuracy improvement compared to the closest competing method. Our method receives RGB images and can directly derive poses in the popular BVH format in real-time allowing a variety of interesting applications and achieving a very good accuracy/performance ratio. We also believe that the techniques described here offer an interesting divide-and-conquer approach that can be generalized and extended to accommodate hand and face pose estimation. This constitutes our next research goal. The research presented, along with its supplementary material and source code are publicly available [54].

Acknowledgments

We gratefully acknowledge the support of NVIDIA Corporation with the donation of a Quadro P6000 GPU used for the execution of this research. This work was partially supported by EU H2020 project Co4robots (Grant No 731869).

[2] https://youtu.be/Jgz1MRq-I-k

References

[1] Y. Chen, Y. Tian, and M. He, "Monocular human pose estimation: A survey of deep learning-based methods," *Computer Vision and Image Understanding*, p. 102897, 2020.

[2] NANSENSE Inc., "Nansense," 2019, [Online; accessed 8-April-2019]. [Online]. Available: https://www.nansense.com/

[3] Oxford Metrics, "Vicon motion capture system," 2019, [Online; accessed 8-April-2019]. [Online]. Available: https://www.vicon.com/

[4] S. Johnson and M. Everingham, "Clustered pose and nonlinear appearance models for human pose estimation." in *bmvc*, vol. 2, no. 4. Citeseer, 2010, p. 5.

[5] Blender Online Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Blender Institute, Amsterdam, 2019. [Online]. Available: http://www.blender.org

[6] MakeHuman Community, "Makehuman," 2019, [Online; accessed 8-April-2019]. [Online]. Available: http://www.makehumancommunity.org/

[7] A. Qammaz and A. A. Argyros, "Mocapnet: Ensemble of snn encoders for 3d human pose estimation in rgb images," in *BMVC*, 2019.

[8] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012, pp. 1097–1105.

[10] A. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural networks," in *IEEE CVPR*, 2014, pp. 1653–1660.

[11] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," *arXiv preprint arXiv:1611.08050*, 2016.

[12] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields," in *arXiv preprint arXiv:1812.08008*, 2018.

[13] G. Rogez, P. Weinzaepfel, and C. Schmid, "Lcr-net: Localization-classification-regression for human pose," in *IEEE CVPR*, 2017, pp. 3433–3441.

[14] ——, "Lcr-net++: Multi-person 2d and 3d pose detection in natural images," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 5, pp. 1146–1161, 2019.

[15] R. Alp Güler, N. Neverova, and I. Kokkinos, "Densepose: Dense human pose estimation in the wild," in *IEEE CVPR*, 2018, pp. 7297–7306.

[16] X. Zhou, M. Zhu, S. Leonardos, K. G. Derpanis, and K. Daniilidis, "Sparseness meets deepness: 3d human pose estimation from monocular video," in *IEEE CVPR*, 2016, pp. 4966–4975.

[17] G. Pavlakos, X. Zhou, K. G. Derpanis, and K. Daniilidis, "Coarse-to-fine volumetric prediction for single-image 3d human pose," in *IEEE CVPR*, 2017, pp. 7025–7034.

[18] B. Tekin, P. Márquez-Neila, M. Salzmann, and P. Fua, "Learning to fuse 2d and 3d image cues for monocular body pose estimation," in *IEEE ICCV*, 2017, pp. 3941–3950.

[19] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik, "End-to-end recovery of human shape and pose," in *IEEE CVPR*, 2018, pp. 7122–7131.

[20] V. Tan, I. Budvytis, and R. Cipolla, "Indirect deep structured learning for 3d human body shape and pose prediction," 2018.

[21] G. Varol, D. Ceylan, B. Russell, J. Yang, E. Yumer, I. Laptev, and C. Schmid, "BodyNet: Volumetric inference of 3D human body shapes," in *ECCV*, 2018.

[22] F. Moreno-Noguer, "3d human pose estimation from a single image via distance matrix regression," in *IEEE CVPR*, 2017, pp. 2823–2832.

[23] X. Sun, J. Shang, S. Liang, and Y. Wei, "Compositional human pose regression," in *IEEE ICCV*, 2017, pp. 2602–2611.

[24] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow, "Harmonic networks: Deep translation and rotation equivariance," in *IEEE CVPR*, 2017, pp. 5028–5037.

[25] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," *arXiv preprint arXiv:1812.07035*, 2018.

[26] B. Wandt, H. Ackermann, and B. Rosenhahn, "A kinematic chain space for monocular motion capture," in *ECCV*, 2018, pp. 0–0.

[27] J. Martinez, R. Hossain, J. Romero, and J. J. Little, "A simple yet effective baseline for 3d human pose estimation," in *IEEE ICCV*, 2017, pp. 2640–2649.

[28] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Trans. on Graphics (TOG)*, vol. 37, no. 4, p. 143, 2018.

[29] M. Meredith, S. Maddock *et al.*, "Motion capture file formats explained," *Department of Computer Science, University of Sheffield*, vol. 211, pp. 241–244, 2001.

[30] A. inc. (2019) The daz-friendly bvh release of cmu motion capture database. Accessed: 2019-10-05. [Online]. Available: https://www.autodesk.com/products/motionbuilder/

[31] S. Li, W. Zhang, and A. B. Chan, "Maximum-margin structured learning with deep networks for 3d human pose estimation," in *IEEE ICCV*, 2015, pp. 2848–2856.

[32] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black, "Keep it smpl: Automatic estimation of 3d human pose and shape from a single image," in *ECCV*. Springer, 2016, pp. 561–578.

[33] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "Smpl: A skinned multi-person linear model," *ACM Trans. on Graphics (TOG)*, vol. 34, no. 6, p. 248, 2015.

[34] A. Qammaz, D. Michel, and A. A. Argyros, "A hybrid method for 3d pose estimation of personalized human body models," in *IEEE Winter Conference on Applications of Computer Vision (WACV 2018)*. IEEE, March 2018. [Online]. Available: http://users.ics.forth.gr/argyros/res_personalizedHumanPose.html

[35] 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," *ACM Transactions on Graphics*, vol. 36, no. 4, 2017. [Online]. Available: http://gvv.mpi-inf.mpg.de/projects/VNect/

[36] T. Yu, Z. Zheng, Y. Zhong, J. Zhao, Q. Dai, G. Pons-Moll, and Y. Liu, "Simulcap: Single-view human performance capture with cloth simulation," *arXiv preprint arXiv:1903.06323*, 2019.

[37] D. Xiang, H. Joo, and Y. Sheikh, "Monocular total capture: Posing face, body, and hands in the wild," in *IEEE CVPR*, 2019, pp. 10 965–10 974.

[38] A. Aristidou and J. Lasenby, "Fabrik: A fast, iterative solver for the inverse kinematics problem," *Graphical Models*, vol. 73, no. 5, pp. 243–260, 2011.

[39] S. Agarwal, K. Mierle, and Others, "Ceres solver," http://ceres-solver.org.

[40] J. J. Moré, "The levenberg-marquardt algorithm: implementation and theory," in *Numerical analysis*. Springer, 1978, pp. 105–116.

[41] M. I. Lourakis *et al.*, "A brief description of the levenberg-marquardt algorithm implemented by levmar," *Foundation of Research and Technology*, vol. 4, no. 1, pp. 1–6, 2005.

[42] J. Konečný, Z. Qu, and P. Richtárik, "Semi-stochastic coordinate descent," *optimization Methods and Software*, vol. 32, no. 5, pp. 993–1005, 2017.

[43] CMU Perceptual Computing Lab, "Openpose output format specifications," 2019, [Online; accessed 9-July-2019]. [Online]. Available: https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master/doc/output.md

[44] Wikipedia contributors, "Euclidean distance matrix — Wikipedia, the free encyclopedia," 2018, [Online; accessed 8-April-2019]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Euclidean_distance_matrix&oldid=862708912

[45] ——, "Euclidean distance matrix — Wikipedia, the free encyclopedia," 2020, [Online; accessed 8-April-2020]. [Online]. Available: https://en.wikipedia.org/wiki/Atan2

[46] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *NIPS*, 2017, pp. 971–980.

[47] P. Baldi and P. J. Sadowski, "Understanding dropout," in *NIPS*, 2013, pp. 2814–2822.

[48] D. Molchanov, A. Ashukha, and D. Vetrov, "Variational dropout sparsifies deep neural networks," in *ICML*. JMLR. org, 2017, pp. 2498–2507.

[49] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[50] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[51] F. Chollet *et al.*, "Keras documentation - model checkpoint," 2015, [Online; accessed 8-April-2019]. [Online]. Available: https://keras.io/callbacks/#modelcheckpoint

[52] B. Hahne. (2010) The daz-friendly bvh release of cmu motion capture database. Accessed: 2018-10-05. [Online]. Available: https://sites.google.com/a/cgspeed.com/cgspeed/motion-capture/daz-friendly-release

[53] C. M. University, "Cmu graphics lab motion capture database," http://mocap.cs.cmu.edu/, 2003, accessed: 2017-06-01.

[54] A. Qammaz, "Mocapnet github repository," 2019, [Online; accessed 11-July-2019]. [Online]. Available: https://github.com/FORTH-ModelBasedTracker/MocapNET

[55] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 7, pp. 1325–1339, 2014.

[56] Y. Du, Y. Wong, Y. Liu, F. Han, Y. Gui, Z. Wang, M. Kankanhalli, and W. Geng, "Marker-less 3d human motion capture with monocular image sequence and height-maps," in *ECCV*. Springer, 2016, pp. 20–36.

[57] I. Kostrikov and J. Gall, "Depth sweep regression forests for estimating 3d human pose from images." in *BMVC*, vol. 1, no. 2, 2014, p. 5.

[58] L. Bo and C. Sminchisescu, "Twin gaussian processes for structured prediction," *International Journal of Computer Vision*, vol. 87, no. 1-2, p. 28, 2010.

[59] H. Yasin, U. Iqbal, B. Kruger, A. Weber, and J. Gall, "A dual-source approach for 3d pose estimation from a single image," in *IEEE CVPR*, 2016, pp. 4948–4956.

[60] X. Zhou, X. Sun, W. Zhang, S. Liang, and Y. Wei, "Deep kinematic pose regression," in *ECCV*. Springer, 2016, pp. 186–201.

[61] B. Tekin, P. Márquez-Neila, M. Salzmann, and P. Fua, "Fusing 2d uncertainty and 3d cues for monocular body pose estimation," *arXiv preprint arXiv:1611.05708*, vol. 2, no. 3, 2016.

[62] M. Sanzari, V. Ntouskos, and F. Pirri, "Bayesian image based 3d pose estimation," in *ECCV*. Springer, 2016, pp. 566–582.

[63] D. Tome, C. Russell, and L. Agapito, "Lifting from the deep: Convolutional 3d pose estimation from a single image," in *IEEE CVPR*, 2017, pp. 2500–2509.

[64] C.-H. Chen and D. Ramanan, "3d human pose estimation= 2d pose estimation+ matching," in *IEEE CVPR*, 2017, pp. 7035–7043.

[65] X. Sun, B. Xiao, F. Wei, S. Liang, and Y. Wei, "Integral human pose regression," in *ECCV*, 2018, pp. 529–545.

[66] N. Sarafianos, B. Boteanu, B. Ionescu, and I. A. Kakadiaris, "3d human pose estimation: A review of the literature and analysis of covariates," *Computer Vision and Image Understanding*, vol. 152, pp. 1–20, 2016.

[67] B. Tekin, X. Sun, X. Wang, V. Lepetit, and P. Fua, "Predicting people's 3d poses from short sequences," *arXiv preprint arXiv:1504.08200*, vol. 2, no. 5, p. 6, 2015.

[68] 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 *2017 International Conference on 3D Vision (3DV)*. IEEE, 2017, pp. 506–516.

[69] J. C. Gower, "Generalized procrustes analysis," *Psychometrika*, vol. 40, no. 1, pp. 33–51, 1975.