

# MocapNET: Ensemble of SNN Encoders for 3D Human Pose Estimation in RGB Images

Ammar Qammaz

[ammarkov@ics.forth.gr](mailto:ammarkov@ics.forth.gr)

Antonis A. Argyros

[argyros@ics.forth.gr](mailto:argyros@ics.forth.gr)

Computer Science Department,

University of Crete,

Heraklion, Crete, Greece

Institute of Computer Science, FORTH,

N. Plastira 100, Vassilika Vouton,

GR70013, Heraklion, Crete, Greece

## Abstract

We present MocapNET, an ensemble of SNN [18] encoders that estimates the 3D human body pose based on 2D joint estimations extracted from monocular RGB images. MocapNET provides an efficient divide and conquer strategy for supervised learning. It outputs skeletal information directly into the BVH [20] format which can be rendered in real-time or imported without any additional processing in most popular 3D animation software. The proposed architecture achieves 3D human pose estimations at state of the art rates of 400Hz using only CPU processing.

## 1 Introduction

Human body pose estimation/recovery (HPR) has received a lot of attention from the computer vision community due to its many important applications. A great volume of research has been carried out using a variety of methodologies [10, 11, 12, 13] but arguably, the biggest advances in the field have been achieved recently thanks to the developments in deep learning and convolutional neural networks. However, despite recent advancements, motion capture (MOCAP) systems still remain dependent on expensive multi-camera setups [45] and cumbersome motion capture suits that feature physical markers to facilitate pose estimation.

Our work presents an effort towards human motion recovery of good quality, which is, nevertheless, achievable at low hardware, setup and operational costs. In contrast to existing methods that try to handle what is essentially the input equivariance problem using extensive architectures, we have chosen an alternate route. Instead of deriving a formulation that treats extraction of joint rotations as the final module of a computationally long chain of discrete steps, we attempt to treat the problem at its core by training a feed-forward network (FNN) that directly regresses joint rotations from 2D input. By decomposing the input and output spaces, we eventually reduce the complexity of the task so that simple and fast to compute FNNs can tackle it. This divide and conquer idea is evident in all of our design choices. The high-dimensional input of localized 2D joints is difficult to be directly correlated to output angles, so it is converted to a richer representation we have named Normalized Signed Distance Matrices (NSDMs). NSDMs (described in Section 3) are an alternate formulation of Euclidean Distance Matrices (EDMs) [12, 13] that in addition to translation invariance

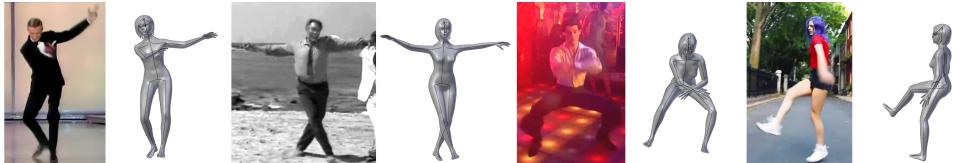


Figure 1: Indicative sample input RGB frames and the corresponding poses recovered by MocapNET. Skinned 3D human body model created using MakeHuman [36].

maintain joint order and are scale resistant, thus simplifying learning. Instead of one big, deep complex and monolithic network, we design separate specialized networks tailored to specific orientations. Each network is further split into smaller specialized encoders, one for each joint.

We train our networks on the publicly available CMU dataset [71]. To deal with its limitations [8, 72], we perturb it, randomize parts of it and simulate self-occlusions to enrich it during data augmentation. Despite the high dimensionality of the 3D pose estimation problem, we still manage to approach it using what is in essence multilayer perceptrons, the simplest neural networks available [59] and among the fastest to compute. At the same time, the powerful self-normalization properties of the SeLU activation layers [28] help reduce their generalization gap. These choices enable 3D pose estimation at state of the art computational performance, i.e. 404.7Hz/2.4ms per evaluation on an Intel Core i7-8700 CPU using Tensorflow [8] 1.14.0. Besides being extremely fast, MocapNET does not rely on an independent kinematic solver, nor any temporal regularisation, and to the best of our knowledge, is the first work that offers direct BVH output from 2D points in an end-to-end neural network.

## 2 Related Work

Human pose estimation is a very active topic with a large volume of recent, novel works.

**2D human body joints estimation:** DeepPose [68] motivated many researchers to incorporate neural networks for the task of 2D pose estimation, moving it away from its initial use as a means for image classification [80, 31]. This trend continued with Tompson et al. [67] that offered a technique that coupled convolutional networks with Markov Random Fields to tackle the problem. This, in turn, revealed the potential of this approach resulting in many 2D pose estimation methods from RGB images that are powered with deep-learning. The stacked hourglass networks of Newell et al. [42], Convolutional Pose Machines [74] and OpenPose [8] are among the most popular works. Other cutting-edge research includes DeepCut [51] as well as the ones presented in [8, 10, 51] which provide the computer vision community with tools to robustly handle the 2D human pose estimation task.

**One-stage 3D human body joints estimation:** Many works that use RGB images as input adopt a holistic approach to the pose estimation problem and infer 3D pose from 2D images in one step. For example, [52] adopts a Bayesian approach, Du et al. [18] defines an intermediate height map generation, Ghezelghieh et al. [20] has a smart camera logic, Rogez et al. [53] uses an image synthesis engine and an end-to-end CNN architecture and LCR-Net [2, 54] combines a pose proposal/classifier with a regressor. Some methods (e.g., DensePose [8], Omran et al. [45] with a combination of a DNN network and the SMPL [55]

model) go a step further by also addressing the problem of human body shape estimation by calculating 3D mesh associations. Kanazawa et.al [26] model the human body using SMPL as well as camera information while also coupling a discriminator network. Chen et. al. [10] employ a 3D pose library and sets of virtual cameras. Tan et al. [62] use pose silhouettes to derive 3D human shape and pose. Other works are focused in acquiring 3D points from 2D images using either a single [4], or more cameras [19, 48]. Some methods use a purely convolutional approach to extract 3D pose like [12] that models 3D volume loss, while some others rely on separate algorithms [8] to perform optimization. Tenkin et.al. [63] uses both RGB to 2D as well as 2D to 3D learning.

**Two-stage 3D human body joints estimation:** Our approach falls to the so called two-stage method category since it separates pose estimation from pattern recognition and operates on 2D points uplifting them to 3D. One-stage methods have the merit of not relying on anything but their own training set. At the same time, this is one of their weaknesses, in the sense that generating an extensive and unbiased dataset is very difficult [11, 54]. Indicatively, there are works specializing in dataset generation using synthetic data coming from 3D models [14] or MOCAP data [3]. High-level data makes data augmentation a much easier task.

Within this method category, Bogo et. al [1] use the SMPL linear model [5] and regress both pose as well as the human shape. Other RGBD based methods [51] use a 3D model acquired using an RGBD camera and use Particle Swarm Optimization [7] as their optimization technique. VNect [40] also uses a two stage RGBD approach but with a generic skeleton model, while Li et al. [34] uses a similar concept albeit using RGB data, only. Certain methods utilize adversarial networks to formulate an inverse graphics problem [55]. Very recent works on RGBD data have reached a point of also accounting for garments [50].

The most similar work to ours is [43] that utilizes 2D EDMs as its input representation, although regressed to 3D EDMs and not to direct output angles like our work. Another recent paper that identifies the need for structure-aware regression is [50] although their joint connection representation is less rich than our NSDM formulation (Sec 3). Our approach shares the compact formulation of [57] although on a much more shallow network without residual connections. Similar approaches include [24] but instead of silhouettes we use the numerical values of 2D points and retrieve results like [49] without physics simulations.

Recent trends show the importance of high-level inference like the one we attempt. Translation and rotation invariant features [70], representations that handle discontinuities in rotations [35] and works that utilize different data mappings that better encode structural properties such as kinematic chain spaces [7] offer promising results. On the other hand advances and improvements on neural networks like dynamically routed neural networks of capsules [56] focus on equivariance by means of architectural changes. Our approach deals with the problem by using a combination of NSDMs (Section 3), independent encoders, and dataset splits that focus on partitions of the pose space. We can thus address a difficult task while using a much less complex feed-forward model formulation.

### 3 Methodology

MocapNET operates on 2D joint input, received in the popular COCO [8] or BODY25 [10, 4] format and internally converted to NSDM matrices. The output is a list of 132 values that correspond to ready-to-use BVH [7] motion frames for a particular input. The first 6 output values for each frame encode skeleton position and orientation, and the rest, angles for each joint of the recovered pose (w.r.t the T-Pose of the armature). Each of the 132 output

parameters is estimated by a standalone encoder trained to accommodate a particular joint. The encoders have a compression parameter  $\lambda$  that controls their size/quality/speed. They are organized in ensembles (Fig. 4) which are themselves organized into classes (Fig. 2), trained to handle particular coarse ranges of human pose orientations.

**Preparing the training and test data:** The original CMU dataset [4] is recorded in ASF/AMC format. However, we use its BVH conversion [2] since it allows direct training on the chosen BVH [4] output format. BVH files can be rapidly deployed to animate skinned human models in 3D editors and game engines (e.g., Blender, Unity, Maya, Lightwave 3D, Cinema 4D, etc). This high-level output can be used to animate any compatibly rigged skinned model. We have chosen the same female model in all illustrations (Figures 1, 6) to showcase that despite the different measurements and even genders of the observed subjects MocapNET does a good job on pose retrieval and a generic parametric MakeHuman [30] model properly reflects the observed poses. Although BVH provides a very compatible output format, the specification also has shortcomings [4]. By using an appropriate armature we overcome most of them and by prepending a T-Pose as the first frame of the file output, we signal the orientation of our armature, since most BVH importer software recognize this as a cue and automatically adjust the BVH skeleton to their internal coordinate system.

Since the BVH standard mandates Euler angles we are forced to utilize them to offer an end-to-end SNN with BVH output. Although Euler angles hinder neural network learning due to discontinuities [25], our class separation scheme (discussed in the next paragraphs) effectively mitigates this problem. We use BVH datasets 01 to 19 from [2] for training. We noticed that datasets 01/01, 01/04, 13/17, 13/18, 13/23, 14/02, 14/03, 14/23, 16/45, 16/46, 16/54, 16/57 and 17/10 contain flipped arm orientations (probably due to errors while porting from the original ASF/AMC dataset) that degrade results. Omitting these problematic datasets leaves us with approximately 484K MOCAP poses, or BVH “motion frames” of depicted actions. Given that acquisition was performed at 120 fps, neighboring frames depict very similar poses. Moreover, skeletons move along the same 3D paths and motions have a high degree of repetitiveness, so the number of available frames appears deceptively large.

**Data augmentation:** To enrich the dataset we create random 2D image locations  $P(x_i, y_i)$  and depth values  $d_i$  where  $0 \leq x_i \leq W$ ,  $0 \leq y_i \leq H$  (in pixels) and  $1000 \leq d_i \leq 5500$  (in mm), assuming video with frame size equal to  $W \times H$ . With these random 2D points and depths and assuming a camera  $C$  with known intrinsic parameters  $W, H, f_x, f_y, c_x, c_y$  we can randomize the skeleton positions of the dataset at 3D points  $(X_i, Y_i, Z_i)$  where  $X_i = (x_i - c_x) \cdot d_i / f_x$ ,  $Y_i = (y_i - c_y) \cdot d_i / f_y$  and  $Z_i = d_i$ . This first data augmentation allows MocapNET to learn how constellations of 2D points are affected by 3D translations.

A second randomization performed on recorded joint angles diversifies captured poses. We perturb the following joints: r/l shoulder, r/l elbow, r/l hand, r/l hip, r/l knee and r/l foot. Their orientations are perturbed by uniform random values so that their new value is at most  $\pm 5^\circ$  away from the original rotation. The perturbation could have been larger, but even using this setting projections shift sufficiently, since rotations stack across the kinematic chain.

Finally, we perform randomization of the orientation of the human skeletons. In order to deal with ambiguities due to symmetries and distinguishing very different poses we also split the randomized poses into three different classes. Assuming a possible rotation vector is  $(r_x, r_y, r_z)$  we define the following orientation classes.

**Class A:** Includes orientations where  $-35^\circ \leq r_x \leq 35^\circ$ ,  $-180^\circ < r_y \leq 180^\circ$  and  $-35^\circ \leq r_z \leq 35^\circ$ . A problem with this class is that around  $r_y 180^\circ, -180^\circ$  (or  $0^\circ, 360^\circ$ ) samples suffer from discontinuities which cause big loss fluctuations for small angle changes during training. We

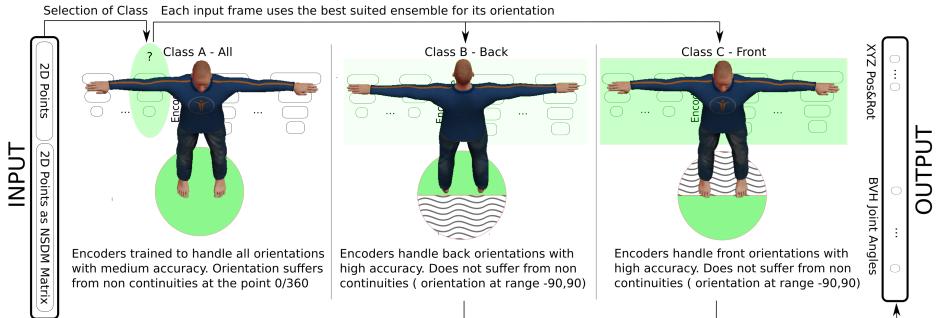


Figure 2: FFNs can't effectively handle direct conversion of 2D points to 3D pose for all possible human orientations at once. Instead of enlarging them we train simpler, faster encoder ensembles specialized in specific body orientations. Class A provides a rough orientation estimation for any input. Using this we choose Class B or C and extract a 3D pose. Each class/ensemble has the same internal organization which is illustrated in Fig 4.

do not use this class for pose estimation but just to decide if an input corresponds to a person facing forward or backward, subsequently picking the correct class for pose retrieval.

**Class B:** Back orientations where  $-35^\circ \leq r_x \leq 35^\circ$ ,  $-90^\circ \leq r_y \leq 90^\circ$  and  $-35^\circ \leq r_z \leq 35^\circ$ .

**Class C:** Front orientations using the same limits as Class B but having the  $r_y$  component shifted by  $180^\circ$  so that  $r_y = 0^\circ$  results in a front facing skeleton (Figure 2).

For each randomized pose we simulate self-occlusions during dataset preparation, by projecting points onto a virtual camera  $C$  and then applying a depth ordering pass that erases joints hidden behind the torso or near the radius of other joints (right example of Figure 3).

We repeat the above randomization process by adding every input pose three times, perturbed differently, to create more training samples that offer a richer source of rotation and translation exemplars. The employed randomization scheme can result in poses where joints fall out of the bounds of the camera. We detect and omit those from our training set. The final training set amounts to  $\approx 1.5M$  training poses per orientation class.

Working with pure MOCAP data and 2D points makes the randomization scheme much more robust than randomization based on appearance [14] since we only use the skeleton information. Since class A ensemble is only exposed to randomized frontal views of the body and class B to randomized back views, our aggregate network is effectively trained on  $\approx 3.0M$  training poses despite never needing to accommodate all of them in GPU memory during training, nor needing to generalize to all of them at once. Our method can be extended to more distinct classes to improve accuracy without computational overheads during evaluation. On the other hand, training time increases linearly with more available classes. Poses with orientations close to class boundaries appear to be gracefully handled using both neighboring classes during experiments. Extra precautions that could be useful in ensembles with many classes is to have overlapping areas between classes to bolster output consistency.

**Representing 2D poses with Normalized Signed Distance Matrices (NSDMs):** Euclidean Distance Matrices (EDMs) [15] offer a simple and useful data representation that encodes data points in relation to each other rather than relative to an external coordinate system. We formulate a variation of EDMs we call Normalized Signed Distance Matrices that are conceptually similar albeit better suited for our task.

The employed COCO [8] or BODY25 [7] 2D joints input consists of a set of  $N$  points

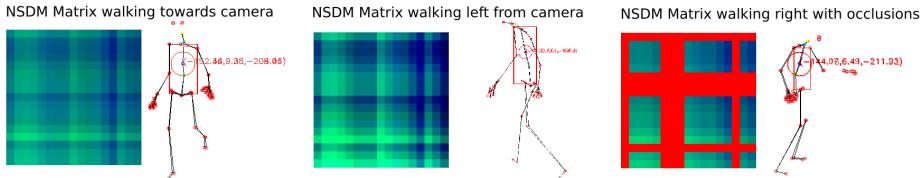


Figure 3: Visualization of the employed NSDM encoding using RGB images where **B** channel encodes pairwise 2D joint distance on the X axis (image width), **G** on Y axis (image height) and **R** occlusions (2D points not present). Although our encoding shares a lot of similarities with EDMs [25] our formulation maintains sign information, is more robust to scale changes and diagonal elements are 0.5 instead of 0.0, except when a joint is occluded.

$J_{2D} = \{p_1, \dots, p_N\}$ . Out of those, we select  $M$  2D points to create NSDM matrices, namely hip, neck, head, l/r collar, l/r shoulder, l/r elbow, l/r hand, l/r hip, l/r knee and l/r foot. We also generate two new artificial points left and right of the hip displaced on the x axis by -0.3 and +0.3 units relative to normalized image width. Since the human body has most of its points along the vertical axis these 2 artificial points contribute with more features on the horizontal axis that better encode small variations in pose that could otherwise be lost.

Each 2D joint  $p_i$  where  $1 \leq i \leq M$  is associated with its coordinates  $p_i = (x_i, y_i)$  which are normalized to the input image frame dimensions and are therefore bounded in the range  $[0, 1]$ . We also associate each such point with a visibility status parameter  $v_i$  provided by thresholding OpenPose joint confidence values (1 if joint is visible, 0 if joint is occluded). We calculate 2 matrices  $NSDM^x$  and  $NSDM^y$ . The entries of the NSDMs are calculated as

$$NSDM^c(i, j) = \begin{cases} 0.5 + c_i - c_j & v_i \neq 0, v_j \neq 0 \\ 0 & \text{otherwise,} \end{cases}$$

for  $c \in \{x, y\}$ . Using this formulation diagonal elements have a value of 0.5 except when occluded. We deviate from the original EDM formulation to allow negative values ( $< 0.5$  in our case) which are useful since they are a source of information differentiating symmetric changes in the configuration of limbs. Our data representation also anticipates the behavior of the SeLU [28] activation function since occluded values fall on its non-linear part. After calculating  $NSDM^x$  and  $NSDM^y$  we calculate the length of the torso and use its value to normalize all matrix elements. This final normalization increases the robustness to scale changes since these affect all limbs. Sample 17x17 NSDMs are visualized in Figure 3.

Instead of convolving kernels in a CNN, by creating input with pairwise association of all inputs we achieve a similar effect in a shallower network with less operations. It is also very important to once again stress that NSDMs are by their definition translation invariant and resistant to scale changes. This, combined with the separation of the human orientation classes (Fig. 2) greatly simplifies the task of learning-based 3D pose estimation.

**Ensemble of SNNs:** Despite the presented preparatory measures, the 2D joints to 3D angles problem remains ill-posed by its definition. Without an explicit camera calibration and limb dimensions as constraints, each 2D point cloud can correspond to arbitrarily many different 3D points and relevant skeleton configurations. On top of this, partially observed skeletons (e.g., due to occlusions) often result in incomplete information. Even in scenes where the full skeleton is observable the employed 2D joints detector [25] can provide noisy estimations. Finally, the high dimensional output means that inevitably we won't be able to densely cover

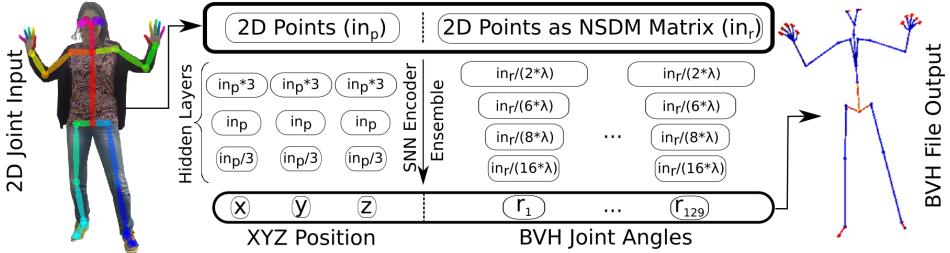


Figure 4: MocapNET SNN encoder ensemble overview for a particular orientation class. We use 3 and 4 layer SNN [☒] encoders that uplift 2D Points to a BVH [☒] motion frame. Value  $\lambda$  is a network compression parameter. Values “ $in_p, in_r$ ” refer to fan-in, the number of input elements of a layer,  $in_p = 171$  elements and for the rest,  $in_r = 578$ .

every possible human body configuration with enough training samples. Self-normalizing Neural Networks (SNNs) [☒] offer a good fit to our problem with their elegant formulation that outperformed other feed forward neural network methods, guaranteeing strong regularization and robust learning. We thus train multiple SNN encoders (illustrated in Fig. 4) that come in two flavors. The first are 3-stage networks designed to work on 2D ( $x, y, v$ ), ( $v$  encodes visibility), input and derive the 3D position for the root bone which is the hip. The second type is 4-stage SNN encoders that regress NSDM input to joint Euler angles in degrees using the corresponding to the rotation of the specific joint.

The 2D body + hands estimations extracted using [☒], form an input array  $in_p$  of 171 elements since we have 57 ( $x, y, v$ ) triplets. Following the steps described in the NSDM elaboration we also get two  $17 \times 17$  NSDM matrices, one for X values and one for Y values that yield  $in_r = 17 \times 17 \times 2 = 578$ . Since all of the 3-stage and 4-stage encoders share inputs we can effectively combine them into an ensemble as seen in Figure 4.

We have identified layer fan-in (the number of input connections) of the layers of the encoders to be one of their most important design considerations since it defines their representational capabilities, how much information flows from layer to layer and ultimately the number of trainable parameters for our model. The scaling parameter  $\lambda$  configures fan-in, as also observed in Fig. 4. Assuming a  $\lambda = 1.5$  we define an ensemble of  $\approx 9.3M$  parameters where each SNN encoder has  $\approx 170K$  parameters.  $\lambda = 1.0$  results to a  $\approx 15M$  ensemble of 270K encoders,  $\lambda = 3.0$  a  $\approx 5M / 70K$  combination. We use  $\lambda = 0.8$  to train the single orientation encoder used from class A and  $\lambda = 1.0$  for the encoders of class B and C.

Networks where  $\lambda > 4$  are much less responsive to observations. Even dated i7-4790 CPUs perform at 2.5ms/387.0Hz for  $\lambda = 2.0$ , a relatively larger  $\lambda = 1.5$  encoder ensemble takes 3.3ms/300.8Hz and a large  $\lambda = 1.0$  ensemble operates at 5.3ms/185.9Hz. Recent i7-8700 CPU hardware performs even better with evaluation for  $\lambda = 1.0$  happening at 2.4ms/404.7Hz,  $\lambda = 1.5$  1.3ms/753.8Hz and  $\lambda = 2.0$  1.2ms/771.7Hz. These rates only account for MocapNET performance (excluding OpenPose [☒] time). Since all tested configurations perform at state of the art rates, we believe that desktop computers should use larger  $\lambda$  values and lower-end or mobile devices can be accommodated with smaller values.

The ensemble approach guarantees a fair share of weights dedicated to each encoder. A hypothetical  $\lambda = 1.0$  ensemble defined as a monolithic 15M network would be much harder to train, offer no way of parameter separation (like the orientation we use on class A) nor any

MocapNET $\lambda = 1$ , Trained on CMU, tested using H36M Blind Protocol 1																
Input	Dir	Dis	Eat	Gre	Pho	Pos	Pur	Sit	Smo	Pho	Wai	Wal	Dog	WaT	Sit.	Avg.
GT	135	140	145	143	153	137	174	215	1565	150	151	156	166	134	246	<b>136</b>
GT + $\mathcal{N}(0,5)$	141	147	150	148	158	141	178	219	161	155	156	159	169	139	248	140
GT + $\mathcal{N}(0,10)$	154	159	162	160	169	152	188	228	174	167	168	170	180	154	256	154
GT + $\mathcal{N}(0,15)$	172	178	180	178	186	170	202	241	190	183	186	187	194	174	267	172
GT + $\mathcal{N}(0,20)$	195	199	198	199	204	189	218	256	208	201	204	207	213	197	280	195

Table 1: Results of MocapNET for  $\lambda = 1.0$  trained on CMU data and tested on Human3.6M using Blind Protocol 1. All numbers are MPJPE in millimeters. We test using Ground Truth (GT) plus different settings of gaussian pixel noise  $\mathcal{N}(\mu, \sigma^2)$  with mean  $\mu$  and variance  $\sigma^2$ . The average error for Protocol 1 is marked with bold.

MocapNET $\lambda = 1$ , Trained on CMU, tested using H36M Blind Protocol 2																
Input	Dir	Dis	Eat	Gre	Pho	Pos	Pur	Sit	Smo	Pho	Wai	Wal	Dog	WaT	Sit.	Avg.
GT	138	145	142	144	151	146	165	194	152	154	150	160	175	138	219	<b>138</b>
GT + $\mathcal{N}(0,5)$	143	150	145	149	155	149	170	197	155	157	154	162	177	142	223	143
GT + $\mathcal{N}(0,10)$	156	163	155	159	164	157	183	205	165	167	170	186	154	230	156	
GT + $\mathcal{N}(0,15)$	175	181	169	175	180	172	197	219	180	182	182	185	196	170	245	175
GT + $\mathcal{N}(0,20)$	194	202	184	194	196	192	216	233	194	199	197	200	215	191	259	194

Table 2: Same as in Table 1 for Blind Protocol 2.

control over what parts of the computational budget affect which joints. Using our ensemble approach, feature selection is specialized for each joint and each loss function during training fully and only reflects each encoder’s assigned task which is a much better proposition.

**Training the ensemble:** We use Keras [15] with the Tensorflow [16] back-end to facilitate training. We use the RMSProp optimizer to train our networks with a learning rate of 0.000025,  $e = 10^{-6}$  and employ a 0.2 dropout in layers trained for 30 epochs. RMSProp is an adaptive learning optimizer which has been criticized by recent works [76], however we found it to be a good choice to facilitate training. The employed loss function uses the mean squared error (MSE) metric to suppress large error deviations and since each of the encoders only handles one parameter, this effectively guides back-propagation during optimization.

We train using mini-batches of 20 samples per batch. This setting balances averaging the loss of the randomized samples without considerable impact on training speed. Higher values gradually lead to less responsive pose tracking while batch-size values less than 12 lead to more jittery output and very long training times. Recent papers [73] advocate the use of a variety of modifications to remedy the problems of large batch sizes but it still remains an open topic. A good overview of the problem is offered by [88]. We suggest batch sizes of 12-32 samples depending on the needs of an application and the available training time.

When starting to train a new class, layers are initialized (as required by SNNs [88]) with 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. However, as we gradually train encoders, neighboring joint angle weights are initialized using weights from the previously closest trained encoder. We employ early training termination by monitoring the loss function and halting if loss improvements are less than 0.001 in 5 or more consecutive training epochs. We also use model checkpoints [16] that monitor loss so that each training session returns the best loss encountered, something which may not necessarily occur during the last epoch.

## 4 Experiments

**Quantitative experiments:** We evaluate MocapNET using the Human 3.6M (H36M) [23] dataset which it is the de facto standard [8, 10, 18, 24, 29, 33, 40, 47, 54, 57, 58, 59, 60, 61,

Comparison of methods tested on H36M Protocol 1 (Method / MPJPE)																
[162]	Our*	[119]	[118]	[116]	[113]	[108]	[107]	[101]	[93]	[88]	[88]	[88]	[82]	[80]	[72]	[40]

Table 3: Comparison of the proposed method to others (errors in mm). MocapNET is only trained in the CMU dataset [70] so its accuracy for P1 is negatively biased.

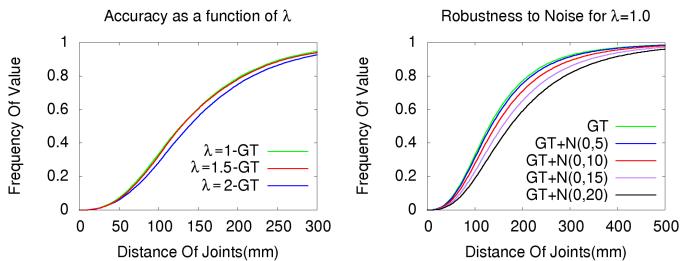


Figure 5: MocapNET accuracy for different  $\lambda$  values (left), and for various levels of Gaussian noise on the 2D input for  $\lambda = 1.0$  (right).

[66, 79, 82, 83]. Evaluation is performed using the mean per joint estimation error (**MPJPE**) for all joints after Procrustes alignment [70] of a method’s estimation to the ground truth. The datasets used for training and evaluation are defined through clearly specified protocols. Protocol 1 dictates training on subjects 1, 4, 6, 7, 8 and performing tests on subjects 9, 11 on 2D points originating from all available cameras. Since we do not train on any data from H36M but otherwise adhere to this protocol, we label it *Blind P1 (BP1)*. Protocol 2 uses the same training and test sets as P1 but only on the frontal camera. We again perform experiments without training on H36M, so we label this as *Blind P2 (BP2)* protocol. There are inconsistencies in the literature about the alignment performed during experiments. For example in [72], P1 uses procrustes alignment and P2 root alignment, whereas in [82], P1 is root and P2 procrustes. We perform both P1 and P2 using procrustes alignment.

Tables 1 and 2 reveal medium accuracy across all actions from all viewing angles with the exception of the two sitting datasets where the topology of the body is more challenging and not well represented in the CMU training examples. This is contrary to other methods which have larger relative fluctuations across actions with the most accurate typically being the walk action. Another interesting result is that on BP2 (Table 2) which reflects frontal view accuracy, and thus an easier case, MocapNET achieves the same average error as in BP1. We attribute this to the employed class structure that effectively decouples orientations. An important result is that the method performs well with noisy input (Fig 5), with an average impact of less than 2cm for a relatively large noise margin ( $\sigma = 10$  pixels).

Our network has never seen people performing many of the H36M actions, nor received any input using similar camera intrinsics, angles etc. Experiments from Yasin et al’s Dual-Source 3D Human Estimation [78] hint at accuracy penalties of  $\approx 30$ mm for our case. The limitations of [70] are very well documented and also one of the main premises of [8] that adds a greater variety of poses with a MOCAP suit. Comparison to other methods is summarized in Table 3. MocapNET is not as accurate as the state of the art but this is natural as it deals with rotation regression and not 3D position regression. Even small errors in each of the joint angles quickly stacks across the kinematic chain with a negative impact on accuracy.

**Qualitative experiments:** We collected various dancing videos from YouTube that feature

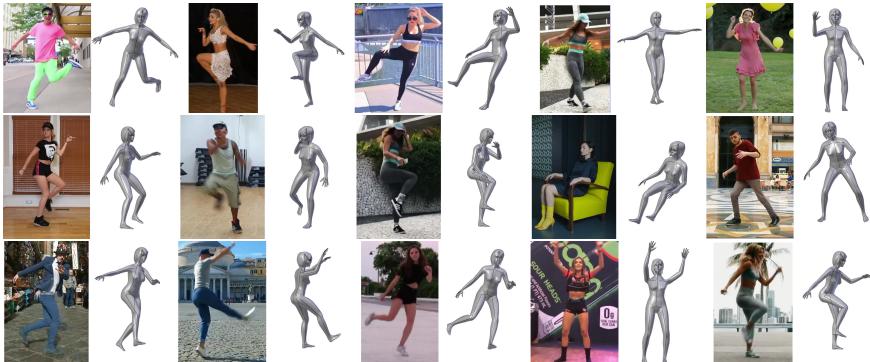


Figure 6: Qualitative results of MocapNET on YouTube dancing videos. Sample input RGB frames and renderings of the estimated poses using a skinned 3D human body model.

interesting/unusual poses, used OpenPose to estimate 2D joints and MocapNET to uplift them to BVH files. We then used MakeHuman [56] to generate a parametric skinned model and Blender [5] to animate the armature. Figure 6 illustrates indicative results. The supplementary material accompanying this paper includes several example output videos. We observe that even if joint positions are not exact (and this also depends on the configurable limb dimensions of the MakeHuman model), motions tend to translate well into the armature. Failure cases mostly consist of persons being very close to the camera and thus typically their feet going out of the field of view. This particular case could be remedied with specialized classes that handle joint estimation without feet. Other problems are caused (a) by poses that are very far from the ones in the training set, (b) by inaccuracies of the 2D joints estimator and (c) by bodies that are curled up and where there is not enough structural information on the NSDM matrices for a clearly defined solution.

## 5 Discussion

We proposed a novel method which, in conjunction with OpenPose, is able to recover 3D human poses from plain RGB input and directly convert them to BVH files. MocapNET provides a fast and efficient baseline method for 3D editing software that allows cheap, easy and direct virtual character animation from pictures or videos. The proposed solution is very fast, conceptually simple and has ample potential for accuracy improvements. Future work should initially extend the training dataset with a richer source of MOCAP data since this seems to currently be the main deficiency. Creating more classes to further decompose input pose space should be attempted since they don't affect performance using our formulation. Adding hands to the training data should also be thoroughly investigated. To encourage research, MocapNET along with its supplementary material can be downloaded at [40].

## Acknowledgements

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 projects Mingei (Grant No 822336) and Co4Robots (Grant No 731869).

## References

- [1] A. Qammaz. Mocapnet github repository, 2019. URL <https://github.com/FORTH-ModelBasedTracker/MocapNET>. [Online; accessed 11-July-2019].
- [2] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL <https://www.tensorflow.org/>. Software available from tensorflow.org.
- [3] Ijaz Akhter and Michael J Black. Pose-conditioned joint angle limits for 3d human pose reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1446–1455, 2015.
- [4] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. Densepose: Dense human pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7297–7306, 2018.
- [5] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam, 2019. URL <http://www.blender.org>.
- [6] Liefeng Bo and Cristian Sminchisescu. Twin gaussian processes for structured prediction. *International Journal of Computer Vision*, 87(1-2):28, 2010.
- [7] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In *European Conference on Computer Vision*, pages 561–578. Springer, 2016.
- [8] Adrian Bulat and Georgios Tzimiropoulos. Human pose estimation via convolutional part heatmap regression. In *European Conference on Computer Vision*, pages 717–732. Springer, 2016.
- [9] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1611.08050*, 2016.
- [10] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. In *arXiv preprint arXiv:1812.08008*, 2018.
- [11] Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4733–4742, 2016.

- [12] 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.
- [13] Lulu Chen, Hong Wei, and James Ferryman. A survey of human motion analysis using depth imagery. *Pattern Recognition Letters*, 34(15):1995–2006, 2013.
- [14] Wenzheng Chen, Huan Wang, Yangyan Li, Hao Su, Zhenhua Wang, Changhe Tu, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Synthesizing training images for boosting human 3d pose estimation. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 479–488. IEEE, 2016.
- [15] Fran ois Chollet et al. Keras. <https://keras.io>, 2015.
- [16] Fran ois Chollet et al. Keras documentation - model checkpoint, 2015. URL <https://keras.io/callbacks/#modelcheckpoint>. [Online; accessed 8-April-2019].
- [17] CMU Perceptual Computing Lab. Openpose output format specifications, 2019. URL <https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master/doc/output.md>. [Online; accessed 9-July-2019].
- [18] Yu Du, Yongkang Wong, Yonghao Liu, Feilin Han, Yilin Gui, Zhen Wang, Mohan Kankanhalli, and Weidong Geng. Marker-less 3d human motion capture with monocular image sequence and height-maps. In *European Conference on Computer Vision*, pages 20–36. Springer, 2016.
- [19] Ahmed Elhayek, Edilson de Aguiar, Arjun Jain, Jonathan Tompson, Leonid Pishchulin, Micha Andriluka, Chris Bregler, Bernt Schiele, and Christian Theobalt. Efficient convnet-based marker-less motion capture in general scenes with a low number of cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3810–3818, 2015.
- [20] Mona Fathollahi Ghezelghieh, Rangachar Kasturi, and Sudeep Sarkar. Learning camera viewpoint using cnn to improve 3d body pose estimation. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 685–693. IEEE, 2016.
- [21] John C Gower. Generalized procrustes analysis. *Psychometrika*, 40(1):33–51, 1975.
- [22] B. Hahne. The daz-friendly bvh release of cmu motion capture database, 2010. URL <https://sites.google.com/a/cgspeed.com/cgspeed/motion-capture/daz-friendly-release>. Accessed: 2018-10-05.
- [23] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In *Advances in Neural Information Processing Systems*, pages 1731–1741, 2017.
- [24] Chaoqun Hong, Jun Yu, Jian Wan, Dacheng Tao, and Meng Wang. Multimodal deep autoencoder for human pose recovery. *IEEE Transactions on Image Processing*, 24(12):5659–5670, 2015.

- [25] 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, 2014.
- [26] Angjoo Kanazawa, Michael J Black, David W Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7122–7131, 2018.
- [27] James Kennedy. Particle swarm optimization. *Encyclopedia of machine learning*, pages 760–766, 2010.
- [28] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. In *Advances in neural information processing systems*, pages 971–980, 2017.
- [29] Ilya Kostrikov and Juergen Gall. Depth sweep regression forests for estimating 3d human pose from images. In *BMVC*, volume 1, page 5, 2014.
- [30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [31] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.
- [32] Subhash Lele. Euclidean distance matrix analysis (edma): estimation of mean form and mean form difference. *Mathematical Geology*, 25(5):573–602, 1993.
- [33] Subhash Lele and Joan T Richtsmeier. Euclidean distance matrix analysis: A coordinate-free approach for comparing biological shapes using landmark data. *American Journal of Physical Anthropology*, 86(3):415–427, 1991.
- [34] Sijin Li, Weichen Zhang, and Antoni B Chan. Maximum-margin structured learning with deep networks for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2848–2856, 2015.
- [35] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):248, 2015.
- [36] MakeHuman Community. Makehuman, 2019. URL <http://www.makehumancommunity.org/>. [Online; accessed 8-April-2019].
- [37] 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.
- [38] Dominic Masters and Carlo Luschi. Revisiting small batch training for deep neural networks. *arXiv preprint arXiv:1804.07612*, 2018.

- [39] 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.
- [40] Dushyant Mehta, Srinath Sridhar, Oleksandr Sotnychenko, Helge Rhodin, Mohammad Shafiei, Hans-Peter Seidel, Weipeng Xu, Dan Casas, and Christian Theobalt. Vnect: Real-time 3d human pose estimation with a single rgb camera. *ACM Transactions on Graphics*, 36(4), 2017. doi: 10.1145/3072959.3073596. URL <http://gvv.mpi-inf.mpg.de/projects/VNect/>.
- [41] Maddock Meredith, Steve Maddock, et al. Motion capture file formats explained. *Department of Computer Science, University of Sheffield*, 211:241–244, 2001.
- [42] Thomas B Moeslund and Erik Granum. A survey of computer vision-based human motion capture. *Computer vision and image understanding*, 81(3):231–268, 2001.
- [43] Francesc Moreno-Noguer. 3d human pose estimation from a single image via distance matrix regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2823–2832, 2017.
- [44] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, pages 483–499. Springer, 2016.
- [45] Mohamed Omran, Christoph Lassner, Gerard Pons-Moll, Peter Gehler, and Bernt Schiele. Neural body fitting: Unifying deep learning and model based human pose and shape estimation. In *2018 International Conference on 3D Vision (3DV)*, pages 484–494. IEEE, 2018.
- [46] Oxford Metrics. Vicon motion capture system, 2019. URL <https://www.vicon.com/>. [Online; accessed 8-April-2019].
- [47] Georgios Pavlakos, Xiaowei Zhou, Konstantinos G Derpanis, and Kostas Daniilidis. Coarse-to-fine volumetric prediction for single-image 3d human pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7025–7034, 2017.
- [48] Georgios Pavlakos, Xiaowei Zhou, Konstantinos G Derpanis, and Kostas Daniilidis. Harvesting multiple views for marker-less 3d human pose annotations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6988–6997, 2017.
- [49] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics (TOG)*, 37(4):143, 2018.
- [50] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter V Gehler, and Bernt Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4929–4937, 2016.

- [51] A. Qammaz, D. Michel, and Antonis 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. URL [http://users.ics.forth.gr/argyros/res\\_personalizedHumanPose.html](http://users.ics.forth.gr/argyros/res_personalizedHumanPose.html).
- [52] Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Reconstructing 3d human pose from 2d image landmarks. In *European Conference on Computer Vision*, pages 573–586. Springer, 2012.
- [53] G. Rogez and C. Schmid. Mocap-guided data augmentation for 3d pose estimation in the wild. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, pages 3108–3116. Curran Associates, Inc., 2016.
- [54] Gregory Rogez, Philippe Weinzaepfel, and Cordelia Schmid. Lcr-net: Localization-classification-regression for human pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3433–3441, 2017.
- [55] Gr  gory Rogez, Philippe Weinzaepfel, and Cordelia Schmid. LCR-Net++: Multi-person 2D and 3D Pose Detection in Natural Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.
- [56] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866, 2017.
- [57] Marta Sanzari, Valsamis Ntouskos, and Fiora Pirri. Bayesian image based 3d pose estimation. In *European conference on computer vision*, pages 566–582. Springer, 2016.
- [58] Nikolaos Sarafianos, Bogdan Boteanu, Bogdan Ionescu, and Ioannis A Kakadiaris. 3d human pose estimation: A review of the literature and analysis of covariates. *Computer Vision and Image Understanding*, 152:1–20, 2016.
- [59] J  rgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.
- [60] Xiao Sun, Jiaxiang Shang, Shuang Liang, and Yichen Wei. Compositional human pose regression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2602–2611, 2017.
- [61] Xiao Sun, Bin Xiao, Fangyin Wei, Shuang Liang, and Yichen Wei. Integral human pose regression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 529–545, 2018.
- [62] Vince Tan, Ignas Budvytis, and Roberto Cipolla. Indirect deep structured learning for 3d human body shape and pose prediction. 2018.
- [63] Bugra Tekin, Xiaolu Sun, Xinchao Wang, Vincent Lepetit, and Pascal Fua. Predicting people’s 3d poses from short sequences. *arXiv preprint arXiv:1504.08200*, 2(5):6, 2015.

- [64] Bugra Tekin, Pablo Márquez-Neila, Mathieu Salzmann, and Pascal Fua. Fusing 2d uncertainty and 3d cues for monocular body pose estimation. *arXiv preprint arXiv:1611.05708*, 2(3), 2016.
- [65] Bugra Tekin, Pablo Márquez-Neila, Mathieu Salzmann, and Pascal Fua. Learning to fuse 2d and 3d image cues for monocular body pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3941–3950, 2017.
- [66] Denis Tome, Chris Russell, and Lourdes Agapito. Lifting from the deep: Convolutional 3d pose estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2500–2509, 2017.
- [67] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in neural information processing systems*, pages 1799–1807, 2014.
- [68] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014.
- [69] Hsiao-Yu Fish Tung, Adam W Harley, William Seto, and Katerina Fragkiadaki. Adversarial inverse graphics networks: Learning 2d-to-3d lifting and image-to-image translation from unpaired supervision. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 4364–4372. IEEE, 2017.
- [70] Carnegie Mellon University. Cmu graphics lab motion capture database. <http://mocap.cs.cmu.edu/>, 2003. Accessed: 2017-06-01.
- [71] Gul Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 109–117, 2017.
- [72] GÜl Varol, Duygu Ceylan, Bryan Russell, Jimei Yang, Ersin Yumer, Ivan Laptev, and Cordelia Schmid. BodyNet: Volumetric inference of 3D human body shapes. In *ECCV*, 2018.
- [73] Bastian Wandt, Hanno Ackermann, and Bodo Rosenhahn. A kinematic chain space for monocular motion capture. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.
- [74] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.
- [75] Wikipedia contributors. Euclidean distance matrix — Wikipedia, the free encyclopedia, 2018. URL [https://en.wikipedia.org/w/index.php?title=Euclidean\\_distance\\_matrix&oldid=862708912](https://en.wikipedia.org/w/index.php?title=Euclidean_distance_matrix&oldid=862708912). [Online; accessed 8-April-2019].
- [76] Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, and Benjamin Recht. The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems*, pages 4148–4158, 2017.

- [77] Daniel E Worrall, Stephan J Garbin, Daniyar Turmukhambetov, and Gabriel J Brostow. Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5028–5037, 2017.
- [78] 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.
- [79] Hashim Yasin, Umar Iqbal, Bjorn Kruger, Andreas Weber, and Juergen Gall. A dual-source approach for 3d pose estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4948–4956, 2016.
- [80] Tao Yu, Zerong Zheng, Yuan Zhong, Jianhui Zhao, Qionghai Dai, Gerard Pons-Moll, and Yebin Liu. Simulcap: Single-view human performance capture with cloth simulation. *arXiv preprint arXiv:1903.06323*, 2019.
- [81] Xiang Yu, Feng Zhou, and Manmohan Chandraker. Deep deformation network for object landmark localization. In *European Conference on Computer Vision*, pages 52–70. Springer, 2016.
- [82] Xiaowei Zhou, Menglong Zhu, Spyridon Leonardos, Konstantinos G Derpanis, and Kostas Daniilidis. Sparseness meets deepness: 3d human pose estimation from monocular video. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4966–4975, 2016.
- [83] Xingyi Zhou, Xiao Sun, Wei Zhang, Shuang Liang, and Yichen Wei. Deep kinematic pose regression. In *European Conference on Computer Vision*, pages 186–201. Springer, 2016.
- [84] Xingyi Zhou, Qixing Huang, Xiao Sun, Xiangyang Xue, and Yichen Wei. Towards 3d human pose estimation in the wild: a weakly-supervised approach. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 398–407, 2017.
- [85] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. *arXiv preprint arXiv:1812.07035*, 2018.