Deep Inertial Poser: Learning to Reconstruct Human Pose from Sparse Inertial Measurements in Real Time

YINGHAO HUANG*, Max Planck Insitute for Intelligent Systems, Tübingen, Germany
MANUEL KAUFMANN*, Advanced Interactive Technologies Lab, ETH Zürich, Switzerland
EMRE AKSAN, Advanced Interactive Technologies Lab, ETH Zürich, Switzerland
MICHAEL J. BLACK, Max Planck Insitute for Intelligent Systems, Tübingen, Germany
OTMAR HILLIGES, Advanced Interactive Technologies Lab, ETH Zürich, Switzerland
GERARD PONS-MOLL, Max Planck Institute for Informatics, Saarbrücken, Germany

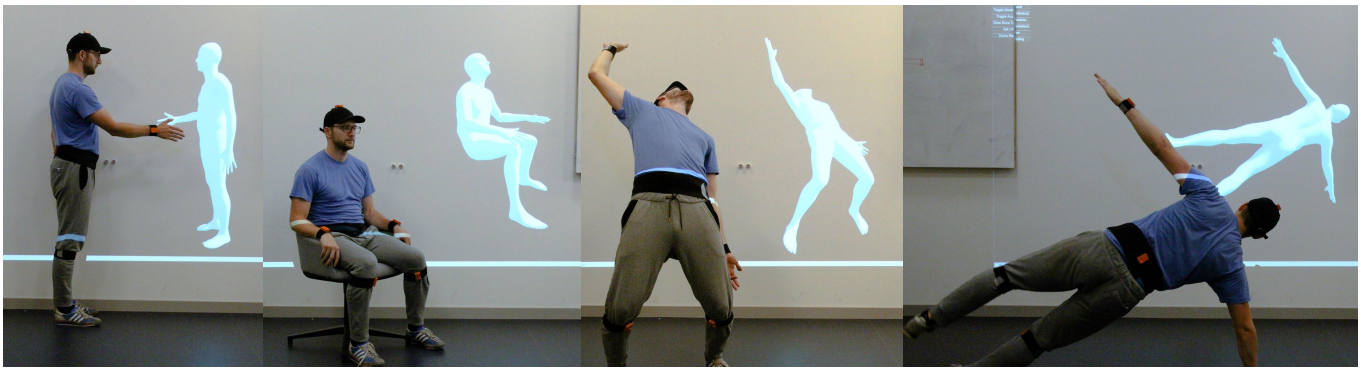

Fig. 1. We demonstrate a novel learning-based method for reconstructing 3D human pose in real-time using only six body-worn IMUs. Our method learns from a large synthetic dataset and at runtime predicts pose parameters from real IMU inputs in real-time while only requiring minimal user instrumentation. This brings 3D motion capture to new scenarios that are difficult for camera-based methods such as heavy occlusions and fast motion.

We demonstrate a novel deep neural network capable of reconstructing human full body pose in real-time from 6 Inertial Measurement Units (IMUs) worn on the user's body. In doing so, we address several difficult challenges. First, the problem is severely under-constrained as multiple pose parameters produce the same IMU orientations. Second, capturing IMU data in conjunction with ground-truth poses is expensive and difficult to do in many target application scenarios (e.g., outdoors). Third, modeling temporal dependencies through non-linear optimization has proven effective in prior work but makes real-time prediction infeasible. To address this important limitation, we learn the temporal pose priors using deep learning. To learn from sufficient data, we synthesize IMU data from motion capture datasets. A bi-directional RNN architecture leverages past and future information that is available at training time. At test time, we deploy the network in a sliding window fashion, retaining real time capabilities. To evaluate our method, we recorded DIP-IMU, a dataset consisting of 10 subjects wearing 17 IMUs for validation in 64 sequences with 330 000 time instants; this constitutes the largest IMU dataset publicly available. We quantitatively evaluate our approach on multiple datasets and show results from a real-time implementation. DIP-IMU and the code are available for research purposes.[1]

CCS Concepts: • **Computing methodologies** → **Motion capture**;

Additional Key Words and Phrases: Real-Time, IMU, Deep Learning, RNN

ACM Reference Format:
Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J. Black, Otmar Hilliges, and Gerard Pons-Moll. 2018. Deep Inertial Poser: Learning to Reconstruct Human Pose from Sparse Inertial Measurements in Real Time. *ACM Trans. Graph.* 37, 6, Article 185 (November 2018), 15 pages. https://doi.org/10.1145/3272127.3275108

1 INTRODUCTION

Many applications such as gaming, bio-mechanical analysis, and emerging human-computer interaction paradigms such as Virtual and Augmented Reality (VR/AR) require a means to capture a user's 3D skeletal configuration. Such applications impose three challenging constraints on pose reconstruction: (i) it must operate in *real-time*, (ii) it should work in everyday settings such as sitting at a desk, at home, or outdoors, and (iii) it should be minimally invasive in terms of user instrumentation.

Most commonly, the task of recording human motion is achieved via commercial motion capture (Mocap) systems such as Vicon[2], but

*Both authors contributed equally to this work

Authors' email addresses: yinghao.huang@tue.mpg.de; manuel.kaufmann@inf.ethz.ch; emre.aksan@inf.ethz.ch; black@tue.mpg.de; otmar.hilliges@inf.ethz.ch; gpons@mpi-inf.mpg.de.

0730-0301/2018/11-ART185
https://doi.org/10.1145/3272127.3275108

[1] http://dip.is.tuebingen.mpg.de
[2] http://www.vicon.com

these require expensive infrastructure and markers placed on the user. Marker-less multi-camera approaches are becoming more accurate and can sometimes provide dense surface reconstructions but also require controlled camera setups and can be computationally very expensive. Recently, the use of a single RGB or RGB-D camera for human pose estimation has become feasible and remains a very active area of research in computer vision [Alldieck et al. 2018; Kanazawa et al. 2018; Mehta et al. 2018; Omran et al. 2018; Tao et al. 2018]. Single camera methods are still less accurate than multi-view methods but more importantly, like all vision-based methods, they require an external camera with the full body visible in the image. This limitation presents a practical barrier for many applications, in particular those where heavy occlusions can be expected such as sitting at a desk or where the user moves around, such as outdoors.

Mounting sensors directly on the user's body overcomes the need for direct line-of-sight. The most prominent choice for pose reconstruction tasks are inertial measurement units (IMUs), which can record orientation and acceleration, are small, and can hence be easily worn on the body. Commercial systems rely on *dense* placement of IMUs, which fully constrain the pose space, to attain accurate skeletal reconstructions [Roetenberg et al. 2007]. Placing 17 or more sensors on the body can be intrusive, time consuming, and prone to errors such as swapping sensors during mounting. However, it has been recently demonstrated, that full body pose can be recovered from a small set of sensors (6 IMUs) [von Marcard et al. 2017], albeit with a heavy computational cost, requiring offline optimization of a non-convex problem over the entire sequence; this can take hours per recording.

Given that emerging consumer products such as smart-watches, fitness trackers and smart-glasses (e.g., HoloLens, Google Glass) already integrate IMUs, reconstructing 3D body pose from a small set of sensors in real-time would enable many applications. In this paper we introduce *DIP: Deep Inertial Poser*, the first deep learning method capable of estimating 3D human body pose from only 6 IMUs in *real time*. Learning a function that predicts accurate poses from a sparse set of orientation and acceleration measurements alone is a challenging task because (i) the whole pose is not observable from just 6 measurements, (ii) previous work has shown that long-range temporal information plays an important role [von Marcard et al. 2017], and (iii) capturing large datasets for training is time-consuming and expensive.

To overcome these issues we leverage the following observations and insights: (i) Large datasets of human Mocap such as CMU [2008] or the H3.6M [2014] exist. Specifically, we leverage AMASS [MoS 2018], a large collection of MoCap datasets with data provided in the form of SMPL [Loper et al. 2015] model parameters. We leveraged this to *synthesize* IMU data. Specifically, we place virtual sensors on the SMPL mesh and use the Mocap sequences to obtain virtual orientations via forward kinematics, and accelerations via finite differences. We leverage this synthetic data for training a deep neural network model. (ii) To model long-range temporal dependencies, we leverage recurrent neural networks (RNNs) to map from orientations and accelerations to SMPL parameters. However, making full use of acceleration information proved to be difficult. This leads to systematic errors for ambiguous mappings between sensor orientation (measured at the lower-extremities) and pose. In particular knee and arm bending is problematic. To alleviate this issue we introduce a novel loss-term that forces the network to reconstruct accelerations during training, which preserves information throughout the network stack and leads to better performance at test time. (iii) Offline approaches for the same task leverage both past and future information [von Marcard et al. 2017]. We propose an extension of our architecture that leverages bi-directional RNNs to further improve the reconstruction quality. At training time, this architecture has access to the same information as [von Marcard et al. 2017], and propagates information from the past to the future and vice versa. To retain the real-time regime, we deploy this architecture at test time in a sliding-window fashion. We experimentally show that only 5 future frames are sufficient for high-quality predictions, while only incurring a modest latency penalty of 85ms.

Using only synthetic data for training already provides decent performance. However, real IMU data contains noise and drift. To close the gap between synthetic and real data distributions, we fine-tune our model using a newly created DIP-IMU dataset containing approximately 90 minutes of real IMU data.

We experimentally evaluate DIP using TotalCapture [Trumble et al. 2017], a benchmark dataset including IMU data and reference ("ground truth") poses, and on the DIP-IMU dataset. We show that DIP achieves an accuracy of 15.85° angular error, which is lower than the competing offline approach, SIP [von Marcard et al. 2017]. This is significant as our method runs in real-time, whereas SIP requires the full motion sequence as input. To further demonstrate the real-time capabilities of DIP, we integrate our approach in a simple VR proof-of-concept demonstrator; we take raw IMU data as input and our pipeline predicts full body poses, without any temporal filtering or post-hoc processing. The resulting poses are then visualized via Unity. In summary, DIP occupies a unique place in the pose estimation literature as it satisfies all three aforementioned constraints: it is real time, minimally intrusive, and works in everyday places Fig. 1.

2 RELATED WORK

The literature on human pose estimation from images and video is vast. Here we briefly review camera-based methods, those that integrate multiple sensor signals using tracking and optimization, and learning based methods that recover pose from sparse sensors.

2.1 Camera-based motion capture:

Commercial camera-based Mocap solutions require that subjects wear many markers and depend on multiple calibrated cameras mounted in the environment. To overcome these constraints, much research has been devoted to developing marker-less approaches from *multiple cameras* (cf. [Sarafianos et al. 2016]). Often such methods require offline processing to achieve high quality results [Ballan et al. 2012; Bregler and Malik 1998; Huang et al. 2017; Starck and Hilton 2003] but recently, real-time approaches [de Aguiar et al. 2008; Elhayek et al. 2017; Rhodin et al. 2015; Stoll et al. 2011] have been proposed. Such approaches typically fit a skeletal model to image data or represent the human as a collection of Gaussians [Stoll et al.

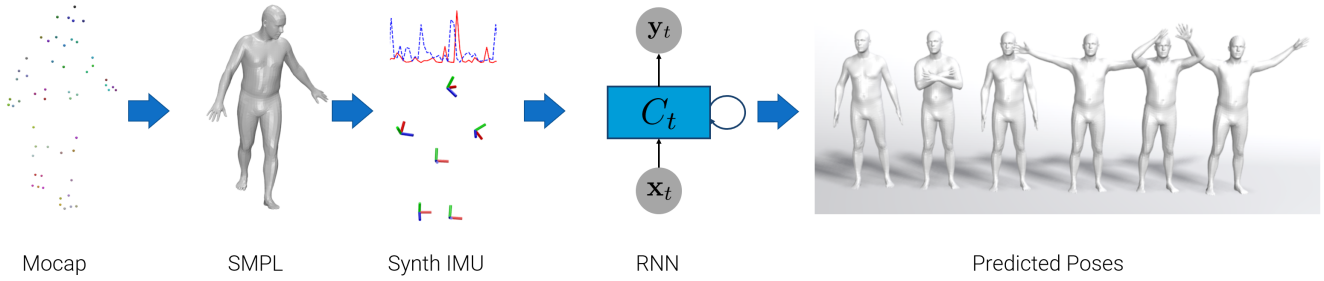

Fig. 2. Overview: *Left*: We leverage existing Mocap datasets to *synthesize* IMU signals from virtual sensors placed on a SMPL mesh. *Middle*: A recurrent neural network takes IMU signals as input and predicts SMPL pose parameters. *Right*: The system runs in real-time and can recover full body pose from just 6 sensors.

2011]. Other approaches to real-time performance include combining discriminative and generative approaches [Elhayek et al. 2017; Oikonomidis et al. 2012]. However, multi-view approaches assume stationary, well calibrated cameras and are therefore not suitable in mobile and outdoor scenarios. More recently pose estimation methods have exploited deep convolutional networks (CNNs) for body-part detection in fully unconstrained *monocular* images [Cao et al. 2016; Chen and Yuille 2014; He et al. 2017; Newell et al. 2016; Tompson et al. 2014; Toshev and Szegedy 2014; Wei et al. 2016]. However, these methods only capture 2D skeletal information. Predicting 3D pose directly from 2D RGB images has been demonstrated using offline methods [Bogo et al. 2016; Tekin et al. 2016; Zhou et al. 2016] and in online settings [Mehta et al. 2018, 2017; Omran et al. 2018]. Using two fish eye cameras worn on the head, pose estimation has been demonstrated [Rhodin et al. 2016]. The setup is still very intrusive for the user, but future miniature cameras could make the approach more practical; such visual data from body-mounted cameras could be complementary to our system. Monocular *depth* cameras provide additional information and have been shown to aid robust skeletal tracking [Ganapathi et al. 2012; Pons-Moll et al. 2015b; Shotton et al. 2013; Taylor et al. 2012; Wei et al. 2012] and enable dense surface reconstruction even under non-rigid deformation [Dou et al. 2016; Newcombe et al. 2015; Tao et al. 2018; Zollhöfer et al. 2014]. Specialized scanners have been used to capture high-fidelity dense surface reconstructions of humans [Collet et al. 2015; Pons-Moll et al. 2017, 2015a].

In contrast to camera-based work, our approach does not rely on calibrated cameras mounted in the environment. Instead, we leverage a sparse set of orientation and acceleration measurements, which makes the pose reconstruction problem much harder but offers the the potential of an infrastructure-free system that can be used in settings where traditional Mocap is not possible.

2.2 Optimization based sensor fusion methods

Inertial trackers. Commercial inertial tracking solutions [Roetenberg et al. 2007] use 17 IMUs equipped with 3D accelerometers, gyroscopes and magnetometers, fused together using a Kalman Filter. Assuming the measurements are noise-free and contain no drift, the 17 IMU orientations completely define the full pose of the subject (using standard skeletal models). However, 17 IMUs are very intrusive for the subject, long setup times are required, and errors such as placing a sensor on the wrong limb are common. To compensate for IMU drift, the pioneering work of Vlasic et al. [2007] uses a custom system with 18 boards equipped with acoustic distance sensors and IMUs. However, the system is also very intrusive and difficult to reproduce.

Video-inertial trackers. Sparse IMUs have also been combined with video input [Malleson et al. 2017; Pons-Moll et al. 2011, 2010; von Marcard et al. 2016], or with sparse optical markers [Andrews et al. 2016] to constrain the problem. Similarly, sparse IMUs have been combined with a depth camera [Helten et al. 2013]; IMUs are only used to query similar poses in a database, which constrain the depth-based body tracker. While powerful, hybrid approaches that use video suffer from the same drawbacks as pure camera-based methods including occlusions and restricted recording volumes. Recent work uses a single moving camera and IMUs to estimate the 3D pose of multiple people in natural scenes [von Marcard et al. 2018], but the approach requires a camera that follows the subjects around.

Optimization from sparse IMUs. Von Marcard et al. [2017] compute accurate 3D poses using only 6 IMUs. They take a generative approach and place synthetic IMUs on the SMPL body model [Loper et al. 2015]. They solve for the sequence of SMPL poses that produces synthetic IMU measurements that best match the observed sequence of real measurements by optimizing over the entire sequence. Like [von Marcard et al. 2017] we also use 6 IMUs to recover full body pose, and we also leverage SMPL. Our approach is however conceptually very different: instead of relying on computationally expensive offline optimization, we learn a direct mapping from sensor data to the full pose of SMPL, resulting in real-time performance and good accuracy despite using only 6 IMUs.

2.3 Learning based methods

Sparse accelerometers and markers. An alternative to sensor fusion and optimization is to learn the mapping from sensors to full body pose. Human pose is reconstructed from 5 accelerometers by retrieving pre-recorded poses with similar accelerations from a database [Slyper and Hodgins 2008; Tautges et al. 2011]. The mapping from acceleration alone to position is however very difficult to learn,

and the signals are typically very noisy. A somewhat easier problem is to predict full 3D pose from a sparse set of markers [Chai and Hodgins 2005]; here online local PCA models are built from the sparse marker locations to query a database of human poses. Good results are obtained since 5-10 marker positions constrain the pose significantly; furthermore the mapping from 3D locations to pose is more direct than from accelerations. This approach requires a multi-camera studio to capture reflective markers.

Motion sensors. Alternatively, position and orientation can be obtained from motion sensors based on inertial-ultrasonic technology. Full pose can be regressed from 6 such sensors [Liu et al. 2011], which provide global orientation *and position*. While global position sensors greatly simplify the inverse problem since measurements are always relative to a static base-station; consequently, capture is restricted to a pre-determined recording volume. Furthermore, such sensors rely on a hybrid inertial-ultrasonic technology, which is mostly used for specialized military applications[3]. Our method uses *only* commercially available IMUs —providing orientation and acceleration but no position, and does not require a fixed base-station.

Sparse IMUs. Learning methods using sparse IMUs as input have also been proposed [Schwarz et al. 2009], where full pose is regressed using Gaussian Processes. The models are trained on specific movements of individual users for each activity of interest, which greatly limits its applicability. Furthermore, Gaussian Processses scale poorly with the number of training samples. Generalization to new subjects and un-constrained motion patterns is not demonstrated.

Locomotion and gait. IMUs are often used for gait analysis and activity recognition, recently in combination with deep learning approaches (cf. [Wang et al. 2017], for example to extract gait parameters via a CNN for medical purposes [Hannink et al. 2016]. Prediction of locomotion has been shown using a single IMU [Mousas 2017] using a hierarchical Hidden Markov Model. Deep learning has been used to produce locomotion of avatars that adapt to irregular terrain [Holden et al. 2017], or that avoid obstacles and follow a trajectory [Peng et al. 2017]. These approaches are suited to cyclic motion patterns, where cycle phase plays a central role.

In summary, existing learning methods either rely on global joint positions as input—which requires external cameras or specialized technology—or are restricted to pre-defined motion patterns. DIP is the first deep-learning method capable of reconstructing full-body motion from a sparse set of sensors in real time.

3 METHOD OVERVIEW

Our goal is to reconstruct articulated human motion in unconstrained settings from a sparse set of IMUs (6 sensors) in *real-time*. This problem is extremely difficult since many parts of the body are not directly observable from the sensor data alone. To overcome this problem, we leverage a state-of-the-art statistical model of human shape and pose, and regress its parameters using a deep recurrent neural network (RNN). In our implementation, we use the SMPL model [Loper et al. 2015] both to synthesize training data and as the output target of the LSTM architecture. This approach ensures that sufficient data is available for training, and encourages that the resulting predictions lie close to the subspace spanned by natural human motion. We now briefly introduce the most salient aspects of the data generation process (Sec. 3.1, Sec. 3.2), the accumulated dataset used for training (Sec. 3.3), and our proposed network architecture (Sec. 3.4). An overview of the entire pipeline can be found in Fig. 2.

3.1 Background: SMPL body model

SMPL is a parametrized model of 3D human body shape and pose that takes 72 pose, and 10 shape, parameters, $\boldsymbol{\theta}$ and $\boldsymbol{\beta}$ respectively, and returns a mesh with $N = 6890$ vertices. Shape and pose deformations are applied to a base template, T_μ, that corresponds to the mean shape a training 3D scans. We summarize [Loper et al. 2015] here for completeness:

$$M(\boldsymbol{\beta}, \boldsymbol{\theta}) = W(T(\boldsymbol{\beta}, \boldsymbol{\theta}), J(\boldsymbol{\beta}), \boldsymbol{\theta}, \mathbf{W}) \quad (1)$$

$$T(\boldsymbol{\beta}, \boldsymbol{\theta}) = T_\mu + B_s(\boldsymbol{\beta}) + B_p(\boldsymbol{\theta}), \quad (2)$$

where W is a linear blend-skinning (LBS) function applied to the template mesh in the rest pose, to which pose- and shape-dependent deformations, $B_p(\boldsymbol{\theta})$ and $B_s(\boldsymbol{\theta})$, are added. The resulting mesh is then posed using LBS with rotations about the joints, $J(\boldsymbol{\beta})$, which depend on body shape. The shape-dependent deformations model subject identity while the pose-dependent ones correct LBS artifacts and capture deformations of the body with pose.

3.2 Synthesizing training data

Our approach is learning-based and hence requires a sufficiently large dataset for training. Compared to the camera or marker-based cases, there are very few publicly available datasets including IMU data and ground-truth poses. To the best of our knowledge the only such dataset is TotalCapture [Trumble et al. 2017], including typical day-to-day activities. The dataset contains synchronized IMU, Mocap and RGB data. However, due to the limited size and types of activities, models trained on this dataset alone do not generalize well, e.g., common interaction tasks in VR/AR such as reaching for and selecting objects in a seated position are not represented at all.

However, given the capability of fitting SMPL parameters to inputs of different modalities (17 IMUs, marker data), it becomes feasible to generate a much larger and more comprehensive training dataset by *synthesizing* pairs of IMU measurements and corresponding SMPL parameters from a variety of input datasets.

To attain synthetic IMU training data, we place *virtual* sensors on the SMPL mesh surface. The orientation readings are then directly retrieved using forward kinematics, whereas we obtain accelerations via finite differences. Assuming the position of a virtual IMU is $\boldsymbol{p}_t$ for time t, and the time interval between two consecutive frames is dt, the simulated acceleration is computed as:

$$\boldsymbol{a}_t = \frac{\boldsymbol{p}_{t-1} + \boldsymbol{p}_{t+1} - 2 * \boldsymbol{p}_t}{dt^2}. \quad (3)$$

3.3 Datasets

Our final training data is a collection of pairs of synthetic IMU sensor readings and corresponding SMPL pose parameters. We use a

[3] http://www.intersense.com/pages/20/14

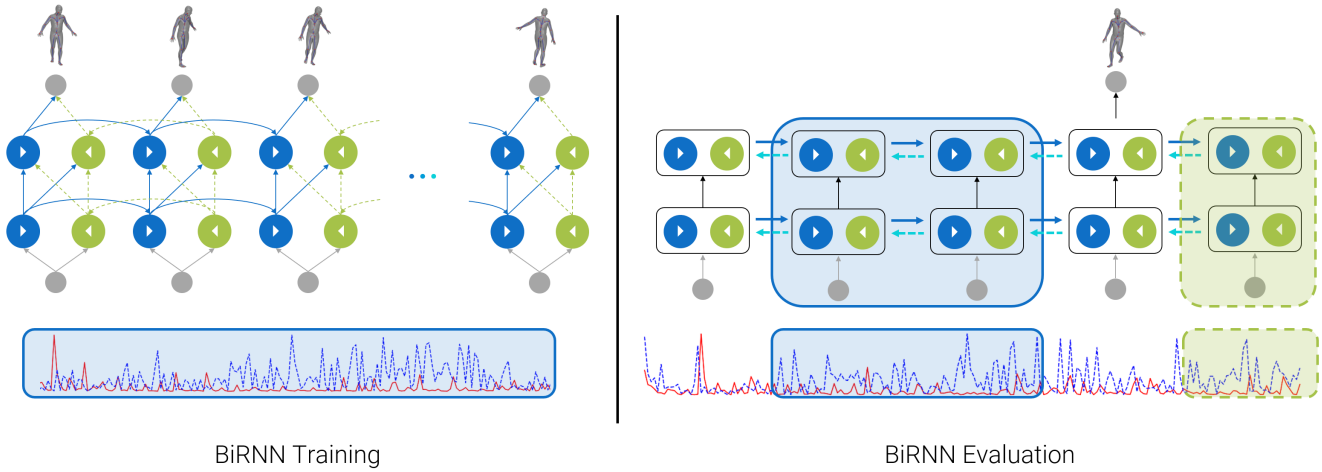

Fig. 3. Overview: *Left*: At training-time our network has access to the whole sequence (blue window) and propagates temporal information from the past to the future and vice versa. Our model consists of two stacked bidirectional layers. Shown in blue (solid arrows) is the forward layer and in green (dashed arrows) the backward layer. Note that the second layer receives direct input from the forward and backward cells of the layer below (diagonal arrows). Please refer to the appendix, Fig. 12, for more details. *Right*: At runtime we feed a sliding window of short subsequences from the past (blue window) and the future (green window) to predict the pose at the current time step. This incurs only minimal latency and makes real-time applications feasible.

subset of the AMASS dataset [MoS 2018], itself a combination of datasets from the computer graphics and vision literature, including CMU [De la Torre et al. 2008], HumanEva [Sigal et al. 2010], JointLimit [Akhter and Black 2015], and several smaller datasets.

We use two further datasets (TotalCapture and DIP-IMU) for evaluation of our method. Both consist of pairs of *real* IMU readings and reference ("ground-truth") SMPL poses. To obtain reference SMPL poses for TotalCapture [Trumble et al. 2017], we used the method of [Loper et al. 2014] on the available marker information. Finally, we recorded the DIP-IMU dataset using commercially available XSens sensors. The corresponding SMPL poses were obtained by running SIP [von Marcard et al. 2017] on all 17 sensors. More details on the data collection is available in Section 4.5.

Note that combining these datasets is non-trivial as most of them use a different number of markers and varying framerates. The datasets involved in this work are summarized in Table 1 . All datasets combined consist of 618 subjects and over 1 million frames of data. To the best of our knowledge no other IMU dataset of this extent is available at the time of writing. We will make the following data available for research purposes: the generated synthetic IMU data on AMASS, and the DIP-IMU dataset–including corresponding ground-truth SMPL poses reconstructed from 17 IMUs and the original IMU data.

3.4 Deep Inertial Poser (DIP)

Given the training dataset $\mathcal{D} = \{(\boldsymbol{x}^i, \boldsymbol{y}^i)\}_{i=1}^N$ consisting of N training sequences, our task is to learn a function $f : \mathbf{x} \rightarrow \mathbf{y}$ that predicts SMPL pose parameters $\mathbf{y}$ from sparse IMU inputs $\mathbf{x}$ (acceleration and orientation for each sensor). This mapping poses a severely underconstrained problem, since there exist potentially many SMPL poses corresponding to the same IMU inputs. For example, consider the case of knee raises while standing in place. Here the orientation data will remain mostly unchanged and only transient accelerations will be recorded throughout the sequence. This observation led to the use of strong priors and a global optimization formulation in [von Marcard et al. 2017], consisting of orientation, acceleration and anthropometric terms. This approach is computationally expensive and offline, with run-times of several minutes to hours depending on the sequence length. To overcome this limitation, we adopt a data-driven approach and model the mapping with neural networks by using a log-likelihood loss function, implicitly learning the space of valid poses from sequences directly.

Both IMU inputs and corresponding SMPL pose targets are highly structured and exhibit strong correlations due to the articulated nature of human motion. Recurrent neural networks are capable of modeling temporal data and have been previously used in modeling of human motion, typically attempting to predict the next frame of a sequence [Fragkiadaki et al. 2015; Ghosh et al. 2017; Martinez et al. 2017]. Although we use a different input modality, our model needs to learn similar motion dynamics. In order to exploit temporal coherency in the motion sequence we use recurrent neural networks (RNN) and bi-directional recurrent neural networks (BiRNN) [Schuster and Paliwal 1997]—with long short-term memory (LSTM) cells [Hochreiter and Schmidhuber 1997].

RNNs summarize the entire motion history via a fixed-length hidden state vector and require the current input $\mathbf{x}_t$ in order to predict the pose vector $\mathbf{y}_t$. While standard RNNs are sufficient in many real-time applications, we experimentally found that having access to both future and past information significantly improves the predicted pose parameters. BiRNNs take all temporal information into account by running two cells in the forward and backward directions, respectively. Compared to RNNs (i.e., unidirectional),

Table 1. Dataset overview. "M" denotes MoCap, "I" denotes IMU and "R" RGB imagery. For details on AMASS see [MoS 2018], for TotalCapture see [Trumble et al. 2017]. Frame numbers and minutes of AMASS correspond to the number and time length of frames we generated at 60 fps by downsampling the original data, where required.

Name	Type	Mode	#Frames	#Minutes	#Subjects	#Motions
AMASS	Synth	M	9,730,526	2703	603	11234
TotalCapture	Real	M, I, R	179,176	50	5	46
DIP-IMU	Real	I	330,178	92	10	64

BiRNNs exhibit better qualitative and quantitative results by accessing the whole input sequence. This is in-line with the findings of [von Marcard et al. 2017] where optimizing over the entire sequence was found to be necessary.

Before arriving at the proposed bidirectional architecture, we experimented with simple feed-forward networks and WaveNet [van den Oord et al. 2016] variants. We found that these models either perform worse quantitatively or produce unacceptable visual jitter. The appendix A contains more details on these experiments. We assume that RNNs can make better use of the inherent temporal properties of the data and hence produce smoother predictions than non-recurrent variants, especially if they have access to future and past information.

While we train our BiRNN model by using all time-steps, it is important to note that at test time, we use only input sub-sequences consisting of past and future frames in a sliding-window fashion. In our real-time processing pipeline, we only permit a short temporal look ahead to keep the latency penalty minimal. Our evaluations show that using only 5 future frames provides the best compromise between performance and latency. Fig. 6 summarizes the impact of window size on the reconstruction quality. We note that in settings with strict low-latency requirements, such as AR, it may be desirable to use no future information at the cost of roughly 1° lower accuracy.

3.4.1 Training with uncertainty. During training, we model target poses $\boldsymbol{y}_t$ with a Normal distribution with diagonal covariance and use a standard log-likelihood loss to train our network:

$$\begin{aligned} \log(p(\mathbf{y})) &= \sum_{t=1}^{T} \log(\mathcal{N}(\mathbf{y}_t|\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t\mathbf{I})), \\ (\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t)_{t=1}^{T} &= f(\mathbf{x}), \end{aligned} \tag{4}$$

where f stands for either a unidirectional or bidirectional RNN being trained on sequences of T frames. In other words, our model f outputs $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ parameters of a Gaussian distribution at every time step. In-line with the RNN literature, we found that this log-likelihood loss leads to slightly better performance than, for example, mean-squared error (MSE).

3.4.2 Reconstruction of acceleration. The input vector for a single frame, $\mathbf{x}_t = [\boldsymbol{o}_t, \boldsymbol{a}_t]$, contains *orientation* $\boldsymbol{o}_t$ and *acceleration* $\boldsymbol{a}_t$ data as measured by the IMUs. We represent orientations as 3×3 rotation matrices in the SMPL body frame. Before feeding orientations and accelerations into the model, we normalize them w.r.t. the root sensor (cf. Section 4.2-4.4). This results in 5 input rotation matrices that are all stacked and vectorized into $\boldsymbol{o}_t = \text{vec}(\{\bar{\mathbf{R}}_1^{TB}, \ldots, \bar{\mathbf{R}}_5^{TB}\}) \in \mathbb{R}^{45}$. Similarly, the normalized accelerations are stacked into $\boldsymbol{a}_t \in \mathbb{R}^{15}$.

The acceleration data $\boldsymbol{a}_t$ is inherently noisy and much less stable than the orientations. This issue is further complicated by the subtle differences between real and synthesized accelerations in the training data. In our experiments, we found that different network architectures displayed the tendency to discard most of the acceleration data already at the input level (almost zero weights on the acceleration inputs). For certain motions, the lack of acceleration information causes the model to underestimate flexion and extension of joints. In order to alleviate this problem, we introduce an auxiliary task during training. Our model is asked to reconstruct the input acceleration in addition to pose at training time. This additional loss forces the model to propagate the acceleration information to the upper layers.

Analogous to the main pose task, we model the auxiliary acceleration loss via a Normal distribution with diagonal covariance.

$$\begin{aligned} \log(p(\boldsymbol{a}_t)) &= \sum_{t=1}^{T} \log(\mathcal{N}(\boldsymbol{a}_t|\boldsymbol{\mu}_{\boldsymbol{a}_t}, \boldsymbol{\sigma}_{\boldsymbol{a}_t}\mathbf{I})), \\ (\boldsymbol{\mu}_{\boldsymbol{a}_t}, \boldsymbol{\sigma}_{\boldsymbol{a}_t})_{t=1}^{T} &= f(\mathbf{x} = [\boldsymbol{o}, \boldsymbol{a}]). \end{aligned} \tag{5}$$

The pose prediction loss Eq. (4) and acceleration reconstruction loss Eq. (5) are complementary to each other and are back-propagated through the architecture with all weights and other network parameters being shared. Only a minimal number of additional trainable network parameters is required to predict $\boldsymbol{\mu}_{a_t}$ and $\boldsymbol{\sigma}_{a_t}$ with sufficient accuracy.

We experimentally show that adding the auxiliary acceleration loss improves pose predictions quantitatively.

3.4.3 Regularization. We train on primarily synthetic data. While the data is sufficiently realistic, slight differences relative to real data are unavoidable. As a consequence, we observed indications of overfitting, and testing on real data yielded less accurate and jerky predictions. To counteract overfitting, we regularize models via dropouts directly on the inputs with a keep probability of 0.8, which randomly filters out 20% of the inputs during training. Randomly masking inputs helps the model to better generalize to the real data and to make smoother temporal predictions.

3.4.4 Fine-tuning with real data. To reduce the gap between real and synthetic data further, we fine-tune the pre-trained models, using the training split of the new dataset (see Sec. 4.5). We found fine-tuning particularly effective in situations where specific usage scenarios or motion types were underrepresented in the training data. Hence, this procedure is an effective means of adapting our method to novel situations.

4 IMPLEMENTATION DETAILS

4.1 Network architecture

We implemented our network architecture in TensorFlow [Abadi et al. 2015]. Fig. 12 in the appendix summarizes the architecture details. We used the Adam optimizer [Kingma and Ba 2014] with an initial learning rate of 0.001, which is exponentially decayed with a rate of 0.96 and decay step 2000. In order to alleviate the exploding

gradient problem, we applied gradient clipping with a norm of 1. We followed the early stopping training scheme by using the validation log-likelihood loss.

4.2 Sensors and calibration

Sensors. We use Xsens IMU sensors [4] containing 3-axis accelerometers, gyroscopes and magnetometers; the raw sensor readings are in the sensor-local coordinate frame F^S. Xsens also provides absolute orientation of each sensor relative to a global inertial frame F^I. Specifically, the IMU readings that we use are *orientation*, provided as a rotation $\mathbf{R}^{IS} : F^S \rightarrow F^I$, which maps from the sensor-local frame to the inertial frame, and *acceleration* which is provided in local sensor coordinates.

Calibration. Before feeding orientation and acceleration to our model, we must transform them to a common body-centric frame, in our case the SMPL body frame F^T. Concretely, we must find the map $\mathbf{R}^{TI} : F^I \rightarrow F^T$, relating the inertial frame to the SMPL body frame. To this end, we place the head sensor onto the head such that the sensor axes align with the SMPL body frame. Consequently, in this configuration, the mapping from head sensor to SMPL frame F^T is the identity. This allows us to set $\mathbf{R}^{TI}$ as the inverse of the orientation $\mathbf{R}_{\text{Head}}$ read from the head sensor at calibration time. All IMU readings can then be expressed in the SMPL body frame:

$$\mathbf{R}_t^{TS} = \mathbf{R}^{TI}\mathbf{R}_t^{IS} = \mathbf{R}_{\text{Head}}^{-1}\mathbf{R}_t^{IS}. \tag{6}$$

Lastly, due to surrounding body tissue, the sensors are offset from the corresponding bones. We denote this constant offset by $\mathbf{R}^{BS} : F^S \rightarrow F^B$, where F^B is the respective bone coordinate frame. In the first frame of each sequence, each subject stands in a known straight pose with known bone orientation $\mathbf{R}_0^{BT}$, and we compute the per-sensor bone offset as:

$$\mathbf{R}^{BS} = \text{inv}(\mathbf{R}_0^{TB})\mathbf{R}_0^{TS}, \tag{7}$$

where $\text{inv}(\cdot)$ denotes matrix inverse.This lets us transform the sensor orientations to obtain *virtual bone orientations* at every frame

$$\mathbf{R}_t^{TB} = \mathbf{R}^{BS}\mathbf{R}_t^{TS}, \tag{8}$$

which we use for training and testing. The interpretation of virtual bone orientations is straightforward: they are the bone orientations as measured by the IMU. The acceleration data is transformed to the SMPL coordinate frame after subtracting gravity, and is denoted as $\boldsymbol{a}$. Calibration only requires the subject to hold a straight pose for a couple of seconds at the beginning of the recording. Fig. 4 provides an overview of the different coordinate frames involved in our calibration process.

4.3 Normalization

For better generalization, the input data should be invariant to the facing direction of the person (e.g. a running motion while the subject is facing north or south should produce the same inputs for our learning model). To this end, we normalize all bone orientations with respect to the root sensor, mounted at the base of the user's spine. With $\mathbf{R}_{\text{root}}(t)$ denoting the orientation of the root at time step t, and $\mathbf{R}_s^{TB}(t)$, the orientation of the bone corresponding to

[4] https://www.xsens.com/

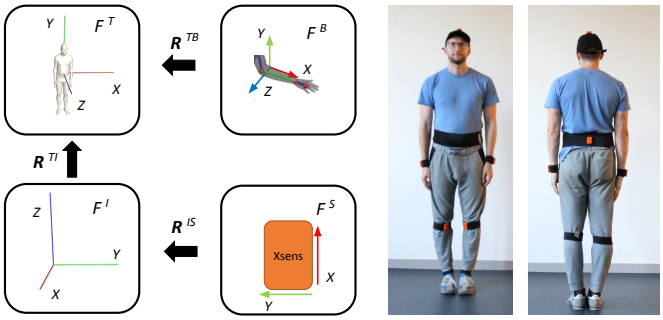

Fig. 4. Calibration overview. *Left:* Overview of coordinate frames. *Right:* Sensor placement and straight pose held by subjects during calibration.

sensor s at time step t, we compute the normalized orientations and accelerations as follows:

$$\bar{\mathbf{R}}_s^{TB}(t) = \mathbf{R}_{\text{root}}^{-1}(t) \cdot \mathbf{R}_s^{TB}(t), \tag{9}$$

$$\bar{\boldsymbol{a}}_s(t) = \mathbf{R}_{\text{root}}^{-1}(t) \cdot (\boldsymbol{a}_s(t) - \boldsymbol{a}_{\text{root}}(t)). \tag{10}$$

This is performed per time-step, both for training and testing. We also experimented with other normalization schemes, such as (i) normalizing the root only w.r.t. the first frame in each sequence and (ii) removing only the heading. We found no significant advantages in either of these schemes compared to the one proposed here. Please refer to the appendix C for details.

4.4 Inputs and targets

The inputs to DIP are the normalized orientations and accelerations. Using 6 IMUs, the input for one frame,

$$\mathbf{x}_t = [\boldsymbol{o}_t, \boldsymbol{a}_t]^T = [\text{vec}(\bar{\mathbf{R}}_1^{TB}(t), \ldots, \bar{\mathbf{R}}_5^{TB}(t)), \bar{\boldsymbol{a}}_1(t), \ldots, \bar{\boldsymbol{a}}_5(t)]^T,$$

is a vector of dimension $d = (3 \cdot 3 + 3) \cdot 5 = 60$. We experimented with more compact representations of orientation than rotation matrices, such as exponential maps or quaternions; these performed significantly worse than using rotation matrices directly. Rotation matrices elements are bounded between $\{-1, 1\}$ which is good for training neural networks. Similarly for the targets $\mathbf{y}$, instead of regressing to the pose parameters $\boldsymbol{\theta}$ of SMPL in axis-angle space, we transform them to rotation matrices and regress them directly. This may seem counter-intuitive because the representation is redundant, but we found empirically that performance is better.

4.5 Data collection

To overcome discrepancies between the sensor data characteristics of the synthetic and real data and to complement the activities portrayed in existing Mocap-based datasets, we recorded an additional dataset of *real* IMU data, which we call DIP-IMU.

We recorded data from 10 subjects (9 male, 1 female) wearing 17 Xsens sensors (see Fig. 4). All the subjects gave informed consent to share their IMU data. To attain ground-truth, we ran SIP [von Marcard et al. 2017] on all 17 sensors. To compensate for different magnetic offsets across IMUs, a heading reset is performed first. The sensors are aligned in a known spatial configuration, after which the heading is reset. Subsequently, the sensors are mounted on the subject and the calibration procedure (cf. Section 4.2) is performed.

Participants were then asked to repeatedly carry out motions in five different categories, including controlled motion of the extremities (arms, legs), locomotion, and also more natural full-body activities (e.g., jumping jacks, boxing) and interaction tasks with everyday objects. In total, we captured approximately 90 minutes of additional data resulting in the largest dataset of real IMU data (appendix F).

5 EXPERIMENTS

To assess the proposed method, we performed a variety of quantitative and qualitative evaluations. We compare our method to the offline baselines SIP [von Marcard et al. 2017] and SOP (reduced version of SIP not leveraging accelerations), and perform self-comparisons between the variants of our architecture. Here we distinguish between two distinct settings. First, we compare performance in the *offline* setting. That is, we use test sequences from existing real datasets (TotalCapture and DIP-IMU). Second, one of the main contributions of our work is the real-time (*online*) capability of our system. To demonstrate this, we implemented an end-to-end live system, taking IMU data as input and predicting SMPL parameters as output.

5.1 Quantitative evaluation

In this section we show how our approach performs on several test datasets. We report both mean joint angle error, computed as in [von Marcard et al. 2017], and positional error.

5.1.1 Offline evaluation. First, we report results from the *offline* setting, in which all models have access to the whole sequence. This setting is a fair comparison between our model and the baselines since SIP and SOP solve an optimization problem over the whole sequence. Table 2 summarizes the results with our models performing close to or better than the SIP baseline. The best configuration (BiRNN (Acc+Dropout)) outperforms SIP by more than one degree. Fig. 5 (left) shows the angular error distribution over the entire TotalCapture dataset. The peak is around 8° error.

The combination of dropouts on the inputs and use of the acceleration loss improve both RNN and BiRNN models. Note that, due to its access to the future steps, BiRNNs perform qualitatively better and produce smoother predictions than the uni-directional RNNs.

5.1.2 Fine-tuning on real data. While the techniques shown in the previous section perform reasonably well on TotalCapture, a significant performance drop is evident on DIP-IMU. This is due to the difference in motions in our new dataset and the aforementioned gap between real and synthetic data distributions. However, the results we have analyzed so far stem from models trained without access to the DIP-IMU data and hence have not seen the types of poses and motions contained in DIP-IMU. We now report the results from our best configurations after fine-tuning on the DIP-IMU data. We fine tune the network on the training split and test it on the held-out set. Tables 2 and 3 show results from the offline and online setting. We find a clear performance increase on DIP-IMU, which is now comparable to TotalCapture. This is further illustrated by the error histogram on DIP-IMU before and after fine-tuning (cf. Fig. 5). Note that the performance on TotalCapture decreases only minimally, indicating that no catastrophic forgetting takes place.

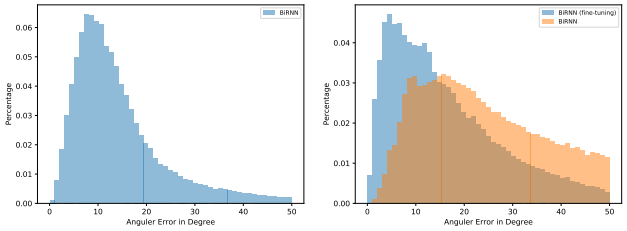

Fig. 5. Histogram of joint angle errors (°). *Left:* Error distribution on TotalCapture with the offline BiRNN model. *Right:* Performance on DIP-IMU before and after fine-tuning as described in Section 5.1.2.

# past frames \ # future frames	0	1	5	10	20	50	100	200
200	16.447	16.108	15.761	15.668	15.649	15.692	15.733	15.752
100	16.417	16.082	15.750	15.652	15.622	15.646	15.675	15.690
50	16.424	16.084	15.768	15.662	15.615	15.619	15.630	15.638
20	16.510	16.177	15.880	15.759	15.684	15.642	15.634	15.622
10	16.578	16.240	15.944	15.815	15.721	15.658	15.638	15.614
5	16.695	16.337	16.009	15.870	15.766	15.690	15.663	15.632
1	17.221	16.683	16.168	15.997	15.883	15.815	15.794	15.768
0	18.008	17.138	16.530	16.346	16.211	16.154	16.150	16.142

Fig. 6. Performance of BiRNN as function of past and future frames on TotalCapture. Numbers are mean joint angle error in degrees. Zero frames means no frames contribute to the prediction from the past or future respectively, i.e. only use the current frame.

5.1.3 Online evaluation. Next, we select our best performing configuration and evaluate it in the online setting. We do not evaluate performance of SIP and SOP since these baselines can not run *online*. Note that now the RNN configurations no longer have access to the entire sequence, but only to past frames in the case of the uni-directional RNN, and to a sliding window of past and a few future frames in the case of the BiRNN. Table 3 indicates that the networks obtain good pose accuracy in the online setting. Notably, the BiRNN with access to 50 past frames even slightly outperforms the offline setting on TotalCapture. This may be due to the accumulation of error in the hidden state of the RNN and the stochasticity of human motion over longer time spans.

In the online setting, the influence of the acceleration loss is most evident. If evaluated on 20 past and 5 future frames on TotalCapture, a BiRNN *without* acceleration loss performs worse ($16.26° \pm 13.54°$) than one using the acceleration loss ($15.88° \pm 13.57°$). For 50 past and 5 future frames the error increases to $16.10° \pm 13.42°$ (compared to $15.77° \pm 13.41°$).

5.1.4 Influence of future window length. Our final implementation leverages a BiRNN to learn motion dynamics from the data. At training time, the entire sequence is processed, but at runtime, only a subset of frames is made available to the network. Fig. 6 summarizes the performance of the network as function of how many frames of past and future information are available at test time. We experimentally found that using 5 future and 20 past frames is the best compromise between prediction accuracy and latency penalty; we use this setting in our live system.

Table 2. Offline evaluation of SOP, SIP, RNN and BiRNN models on TotalCapture [Trumble et al. 2017] and DIP-IMU. Errors reported as joint angle errors in degrees and positional erros in centimeters. Models with *Dropout* are trained by applying dropout on input sequences. *Acc* corresponds to acceleration reconstruction loss. SOP, SIP and BiRNN have access to the whole input sequence while RNN models only use inputs from the past. BiRNN (after fine-tuning) is BiRNN (Acc+Dropout) fined-tuned on DIP-IMU using acceleration reconstruction loss and dropout as well.

	TotalCapture				DIP-IMU			
	μ_{ang}[deg]	σ_{ang}[deg]	μ_{pos}[cm]	σ_{pos}[cm]	μ_{ang}[deg]	σ_{ang}[deg]	μ_{pos}[cm]	σ_{pos}[cm]
SOP	22.18	17.34	8.39	7.57	27.78	19.50	8.23	6.74
SIP	16.98	13.26	5.97	5.50	24.00	16.91	6.34	5.86
RNN (Dropout)	16.83	13.41	6.27	6.32	35.66	19.96	13.38	8.84
RNN (Acc)	16.07	13.16	6.06	6.01	41.00	29.36	15.30	12.96
RNN (Acc+Dropout)	16.08	13.46	6.21	6.27	30.90	18.66	11.84	8.59
BiRNN (Dropout)	15.86	13.12	6.09	6.01	34.55	19.62	12.85	8.62
BiRNN (Acc)	16.31	12.28	**5.78**	5.62	37.88	24.68	14.31	11.30
BiRNN (Acc+Dropout)	**15.85**	12.87	5.98	6.03	31.70	17.30	12.07	8.72
BiRNN (after fine-tuning)	16.84	13.22	6.51	6.17	**17.54**	11.54	**6.49**	5.36

Table 3. Online evaluation of BiRNN models on TotalCapture [Trumble et al. 2017] and DIP-IMU. We select the best performing model from our offline evaluation, i.e., (Acc+Dropout). Numbers in brackets (x, y) mean that this model is evaluated in online mode using x past and y future frames. (fine-tuning) means that the model was fine-tuned on DIP-IMU.

	TotalCapture				DIP-IMU			
	μ_{ang}[deg]	σ_{ang}[deg]	μ_{pos}[cm]	σ_{pos}[cm]	μ_{ang}[deg]	σ_{ang}[deg]	μ_{pos}[cm]	σ_{pos}[cm]
BiRNN (20, 5)	15.88	13.57	6.00	6.16	38.42	25.06	14.49	11.42
BiRNN (50, 5)	**15.77**	13.41	**5.96**	6.13	39.11	24.70	14.81	11.52
BiRNN (fine-tuning)	16.84	13.22	6.51	6.17	**17.54**	11.54	**6.49**	5.36
BiRNN (20, 5) (fine-tuning)	16.90	13.83	6.46	6.26	18.49	12.88	6.63	5.54
BiRNN (50, 5) (fine-tuning)	16.74	13.64	6.42	6.22	18.14	12.75	6.52	5.48

Fig. 7. Selected frames from Playground dataset.

5.2 Qualitative evaluation

We now further assess our approach via qualitative evaluations. Based on the above quantitative results, we only report results from our best model (BiRNN). We use the model in offline mode to produce the results in this section. Section 5.3 discusses the results when using it in online mode. Please see the accompanying video.

5.2.1 Playground (real). First, we compare to SIP and SOP on the Playground dataset [von Marcard et al. 2017]. Playground is challenging because it is captured outdoors and contains uncommon poses and motions. Since the dataset contains no ground truth, we provide only qualitative results. Fig. 7 shows selected frames from a sequence where the subject climbs over an obstacle. We find that SOP has a lot of trouble in reconstructing the leg motion and systematically underestimates arm and knee bending. Our results are comparable to SIP although sometimes the limbs are more outstretched than in the baseline. However, note that SIP optimizes over the whole sequence and is hence computationally very expensive, whereas ours produces predictions in milliseconds.

5.2.2 TotalCapture (real). Here we provide a qualitative comparison of our method and the baseline on the TotalCapture dataset [Trumble et al. 2017]. Fig. 8 summarizes three different sample frames from the dataset. We note that for challenging motions such as back bending (bottom row) and leg raises (middle row), our model outperforms both SIP and SOP and is very close to the reference. Fig. 8 also shows a case where our model successfully reconstructs a leg-raise when SIP and SOP fail. Note however, that this difficult motion also fails to be reconstructed by our model at times (cf. Section 6.2).

5.2.3 DIP-IMU (real). Lastly, we illustrate results on our own DIP-IMU dataset (cf. Sec. 3.3 and Appendix F). Here the reference ("ground truth") is obtained by running SIP using all 17 IMUs. At test time, however, we only use 6 IMUs as input for our method, and

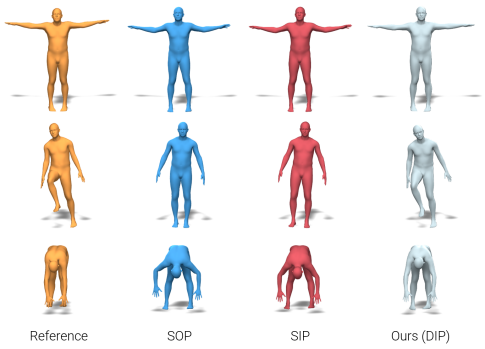

Fig. 8. Sample frames from TotalCapture data set (S1, ROM1).

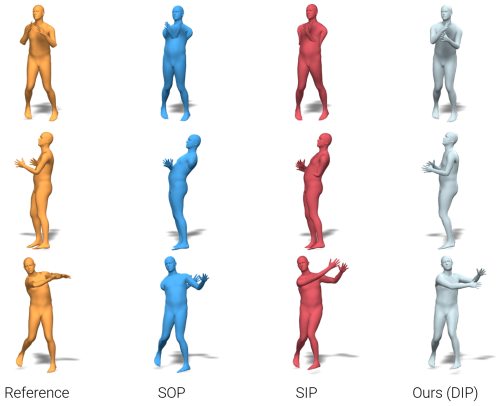

Fig. 9. Sample frames from DIP-IMU (S10, Motion4).

for the baselines SIP and SOP. Fig. 9 summarizes several sequences from the dataset. It is evident that our model outperforms both SIP and SOP qualitatively in a consistent way. We see that SIP/SOP creates a lot of inter-penetrations between limbs and the torso. Our model more faithfully reproduces the arm motions over a large range of frames and poses. Interestingly, our model rarely produces inter-penetrations and produces smooth motion despite noise in the inputs (see video) and without any explicit smoothness or inter-penetration constraints. Hence, DIP learns a mapping from IMU data to the space of valid poses and motion. Smoothness may be explained by the regularization of training via dropouts.

5.3 Live demo

To demonstrate that our model runs in real-time, we implemented an on-line system that streams the sensor data directly into the model and displays the output poses. The raw IMU readings are retrieved via the Xsens SDK, the model's output is displayed via Unity and all communication is performed via the network. Example frames from the live demo are shown in Fig. 1 and Fig. 10. The results are best viewed in the supplementary video. For the live demo we use the online version of the fine-tuned BiRNN model as explained in Section 5.2, i.e. using 20 frames from the past and 5 from the future. The system runs at approximately 29 fps while still producing faithful results (cf. appendix E).

6 DISCUSSION AND LIMITATIONS

6.1 Generalization

In this paper we have shown that our model is able to generalize well to unseen data based on the following observations. (i) We achieve good results on a held-out dataset with real IMU recordings (Total Capture) despite training on synthetic data only (cf. Section 5.1). (ii) We show good qualitative performance on another held-out dataset, Playground, and in the live demo (cf. Section 5.2). This is challenging to achieve due to differences in motions, sensors, and data preprocessing across the datasets. (iii) The system is robust w.r.t. different root orientations (cf. Fig. 1). However, robustness to even more poses, datasets and settings is still the subject of future work. We hypothesize that one of the main limitations is the difficulty of modeling accelerations (synthetic and real) effectively. This is the main reason for fine-tuning on DIP-IMU, which improves generalization but certainly does not have to be the final solution to this problem. In the following we report additional experiments to provide insight into these issues.

Synthetic vs. real. We first train a BiRNN on a smaller, real dataset (DIP-IMU) as opposed to a large, synthetic one (AMASS). We subsequently evaluate this model on TotalCapture, where we notice a drop in performance of around 5.2° (21.03° ±16.35°). Testing on the DIP-IMU held-out set yields an error of (18.84° ±14.08°), which is comparable to the performance when training with synthetic and fine-tuning with DIP-IMU (17.54°). However, the latter yields better performance on TotalCapture. These results illustrate the benefits of a large synthetic motion database.

Re-Synthesis. To analyze the impact of differences between synthetic and real data, we synthesize the IMU measurements for the real datasets (TotalCapture and DIP-IMU). We then evaluate our best BiRNN on these synthetic versions of TotalCapture and DIP-IMU and compare it with the performance on real data. The model performs better on the synthetic version of both TotalCapture (improvement by 2.71° to 13.14° ±10.50°) and DIP-IMU (improvement by 8.84° to 22.86° ±15.70°), highlighting domain differences that need to be addressed. Appendix D provides a more detailed discussion and additional experiments. In summary, we hypothesize that differences in accelerations lie at the core of this problem.

6.2 Failure cases

Fig. 11 (top) shows typical failure cases, taken from an example sequence in TotalCapture. While the model is robust to various root orientations in the live demo, extreme cases, where the body is parallel to the floor (such as push-ups), are challenging. Finally, we compute the worst 5% poses on the test set of DIP-IMU, which results in a mean joint angle error of 43.68° ±8.53° degrees and show the worst poses in appendix B. Leg raises are among the most difficult motions as the sensors show very similar orientation readings while performing this motion. Hence, only acceleration can disambiguate these [von Marcard et al. 2017]. However, sensitivity

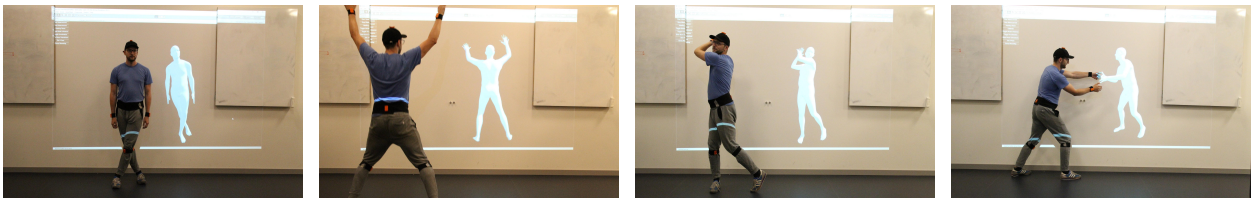

Fig. 10. Sample frames from the live demo showing that our model is able to handle various motion types. See also Fig. 1.

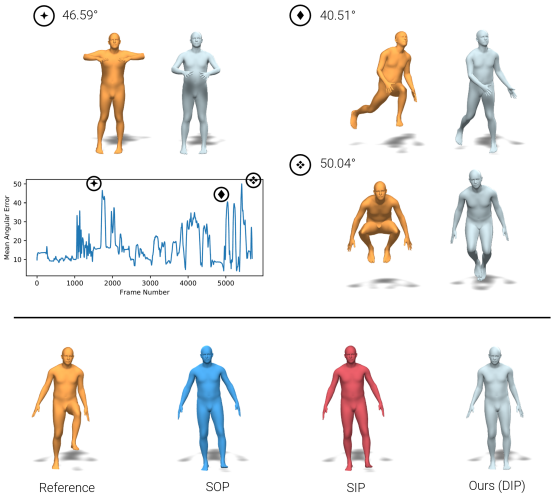

Fig. 11. *Top:* Per-frame average angular error over a sequence from TotalCapture (S3, ROM1) including 3 maximum error poses. The mean angular error for this sequence is $16.73^\circ \pm 8.55^\circ$. *Bottom:* Typical failure case from TotalCapture. Our model and both baselines fail to reconstruct the leg raise.

to IMU placement, environmental factors, and different noise characteristics per sensor make it extremely challenging to integrate them effectively into a learning-based approach. Fig. 11 (bottom) shows how both our model and the baselines fail to reconstruct a leg raise. We believe that these problems arise from the fact that our model struggles to fully exploit the acceleration information. Hence, future work should focus on addressing this challenge by modeling the noise in acceleration and exploring new sensor placement.

7 CONCLUSION AND FUTURE WORK

We presented a new learning-based pose-estimation method that requires only 6 IMUs as input, runs in real-time, and avoids the direct line-of-sight requirements of camera-based systems. We leverage a large Mocap corpus to synthesize IMU data (orientation and acceleration) using the SMPL body model. From this synthetic data, we show how to learn a model (DIP) that generalizes to real IMU data, obtaining an accuracy of 15.85° angular error on TotalCapture. We exploit temporal information by using a bi-directional RNN that propagates information forward and backwards in time; at training time DIP has access to full sequences, whereas at test time the model has access to the last 20 frames and only 5 frames in the future. This produces accurate pose estimates at a latency of only 85ms. Even satisfying the real-time requirement, DIP performs comparably to, or better than, the competing off-line approach, SIP. Furthermore, DIP produces results that are smooth and generally without interpenetrations. This demonstrates that DIP learns a mapping to the space of valid human poses without requiring explicit smoothness or joint angle limits.

Future work should address capturing multi-person interactions and people interacting with objects and the environment. While the focus of this work has been a system based purely on wearable sensors, some applications admit external, or body mounted cameras [Rhodin et al. 2016]. It would be interesting to integrate visual input with our tracker in order to obtain even better pose estimates, especially to capture contact points, knee bends and sitting-down poses, which are difficult to recover using only 6 IMUs. While our approach runs in real-time, transferring the motion data over the Internet may introduce latency, which is a problem for virtual social interaction. Hence, we will explore ways to predict into the future to reduce latency. Finally, unlike [von Marcard et al. 2017], no global translation is considered in our method. This limitation can be critical in some application scenarios. We see two possible solutions to this. First, a GPS signal, which is integrated into most phones, could be integrated into DIP to obtain reasonable global position. Another potential way is to regress the global translations directly from the temporal IMU inputs. We leave this for future work.

We have demonstrated the capabilities of DIP by displaying its pose predictions in real time. We believe that real-time pose estimation methods, which require only a small number of wearable sensors like DIP, will play a key role for emerging interactive technologies such as VR and AR.

ACKNOWLEDGMENTS

We would like to express our gratitude to Timo von Marcard for providing code and support to run SIP/SOP and for fruitful discussions throughout this project. We thank the reviewers for their valuable comments and all participants for their efforts spent in recording DIP-IMU. We are also grateful to Velko Vechev for his extensive help with the live demo and Jonathan Williams and Benjamin Pellkofer for web development. This work was supported in part by the ERC Grant OPTINT (StG-2016-717054). We thank the NVIDIA Corporation for the donation of GPUs used in this work. **Disclosure:** MJB has received research gift funds from Intel, Nvidia, Adobe, Facebook, and Amazon. While MJB is a part-time employee of Amazon, his research was performed solely at, and funded solely by, MPI.

REFERENCES

2018. Unifying Motion Capture Datasets by Automatically Solving for Full-body Shape and Motion. *In preparation* (2018).

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. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. (2015). https://www.tensorflow.org/ Software available from tensorflow.org.

Ijaz Akhter and Michael J Black. 2015. Pose-conditioned joint angle limits for 3D human pose reconstruction. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition*. IEEE, 1446–1455.

Thiemo Alldieck, Marcus Magnor, Weipeng Xu, Christian Theobalt, and Gerard Pons-Moll. 2018. Video Based Reconstruction of 3D People Models. In *IEEE Conference on Computer Vision and Pattern Recognition*. CVPR Spotlight.

Sheldon Andrews, Ivan Huerta, Taku Komura, Leonid Sigal, and Kenny Mitchell. 2016. Real-time Physics-based Motion Capture with Sparse Sensors. In *Proceedings of the 13th European Conference on Visual Media Production (CVMP 2016)*. ACM, 5.

Luca Ballan, Aparna Taneja, Jürgen Gall, Luc Van Gool, and Marc Pollefeys. 2012. Motion capture of hands in action using discriminative salient points. *Computer Vision–ECCV 2012* (2012), 640–653.

Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J. Black. 2016. Keep it SMPL: Automatic Estimation of 3D Human Pose and Shape from a Single Image. In *Computer Vision – ECCV 2016 (Lecture Notes in Computer Science)*. Springer International Publishing, 561–578.

Christoph Bregler and Jitendra Malik. 1998. Tracking people with twists and exponential maps. In *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*. IEEE, 8–15.

Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2016. Realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1611.08050* (2016).

Jinxiang Chai and Jessica K Hodgins. 2005. Performance animation from low-dimensional control signals. In *ACM Transactions on Graphics (TOG)*, Vol. 24. ACM, 686–696.

Xianjie Chen and Alan L Yuille. 2014. Articulated pose estimation by a graphical model with image dependent pairwise relations. In *NIPS*. 1736–1744.

Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. 2015. High-quality streamable free-viewpoint video. *ACM Transactions on Graphics* 34, 4 (2015), 69.

Edilson de Aguiar, Carsten Stoll, Christian Theobalt, Naveed Ahmed, Hans-Peter Seidel, and Sebastian Thrun. 2008. Performance Capture from Sparse Multi-view Video. In *ACM SIGGRAPH 2008 Papers (SIGGRAPH '08)*. ACM, New York, NY, USA, Article 98, 10 pages. https://doi.org/10.1145/1399504.1360697

Fernando De la Torre, Jessica Hodgins, Adam Bargteil, Xavier Martin, Justin Macey, Alex Collado, and Pep Beltran. 2008. Guide to the carnegie mellon university multimodal activity (cmu-mmac) database. *Robotics Institute* (2008), 135.

Mingsong Dou, Sameh Khamis, Yury Degtyarev, Philip Davidson, Sean Ryan Fanello, Adarsh Kowdle, Sergio Orts Escolano, Christoph Rhemann, David Kim, Jonathan Taylor, Pushmeet Kohli, Vladimir Tankovich, and Shahram Izadi. 2016. Fusion4D: Real-time Performance Capture of Challenging Scenes. *ACM Trans. Graph.* 35, 4, Article 114 (July 2016), 13 pages. https://doi.org/10.1145/2897824.2925969

Ahmed Elhayek, Edilson de Aguiar, Arjun Jain, J Thompson, Leonid Pishchulin, Mykhaylo Andriluka, Christoph Bregler, Bernt Schiele, and Christian Theobalt. 2017. MARCOnIâĂŤConvNet-Based MARker-Less Motion Capture in Outdoor and Indoor Scenes. *IEEE transactions on pattern analysis and machine intelligence* 39, 3 (2017), 501–514.

Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. 2015. Recurrent network models for human dynamics. In *Computer Vision (ICCV), 2015 IEEE International Conference on*. IEEE, 4346–4354.

Varun Ganapathi, Christian Plagemann, Daphne Koller, and Sebastian Thrun. 2012. Real-time human pose tracking from range data. In *European conference on computer vision*. Springer, 738–751.

Partha Ghosh, Jie Song, Emre Aksan, and Otmar Hilliges. 2017. Learning Human Motion Models for Long-term Predictions. *arXiv preprint arXiv:1704.02827* (2017).

Julius Hannink, Thomas Kautz, Cristian Pasluosta, Karl-Gunter Gassmann, Jochen Klucken, and Bjoern Eskofier. 2016. Sensor-based Gait Parameter Extraction with Deep Convolutional Neural Networks. *IEEE Journal of Biomedical and Health Informatics* (2016), 85–93.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE, 2980–2988.

Thomas Helten, Meinard Muller, Hans-Peter Seidel, and Christian Theobalt. 2013. Real-time body tracking with one depth camera and inertial sensors. In *Proceedings of the IEEE International Conference on Computer Vision*. 1105–1112.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

Daniel Holden, Taku Komura, and Jun Saito. 2017. Phase-functioned neural networks for character control. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 42.

Yinghao Huang, Federica Bogo, Christoph Lassner, Angjoo Kanazawa, Peter V. Gehler, Javier Romero, Ijaz Akhter, and Michael J. Black. 2017. Towards Accurate Marker-less Human Shape and Pose Estimation over Time. In *International Conference on 3D Vision (3DV)*. 421–430.

Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. 2014. 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 (jul 2014), 1325–1339.

Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik. 2018. End-to-end Recovery of Human Shape and Pose. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

Huajun Liu, Xiaolin Wei, Jinxiang Chai, Inwoo Ha, and Taehyun Rhee. 2011. Realtime human motion control with a small number of inertial sensors. In *Symposium on Interactive 3D Graphics and Games*. ACM, 133–140.

Matthew Loper, Naureen Mahmood, and Michael J Black. 2014. MoSh: Motion and shape capture from sparse markers. *ACM Transactions on Graphics (TOG)* 33, 6 (2014), 220.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. 2015. SMPL: A skinned multi-person linear model. *ACM Transactions on Graphics (TOG)* 34, 6 (2015), 248.

Charles Malleson, Marco Volino, Andrew Gilbert, Matthew Trumble, John Collomosse, and Adrian Hilton. 2017. Real-time Full-Body Motion Capture from Video and IMUs. In *2017 Fifth International Conference on 3D Vision (3DV)*. 449–457.

Julieta Martinez, Michael J Black, and Javier Romero. 2017. On human motion prediction using recurrent neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 4674–4683.

Dushyant Mehta, Oleksandr Sotnychenko, Franziska Mueller, Weipeng Xu, Srinath Sridhar, Gerard Pons-Moll, and Christian Theobalt. 2018. Single-Shot Multi-Person 3D Pose Estimation From Monocular RGB. In *International Conference on 3D Vision (3DV)*.

Dushyant Mehta, Srinath Sridhar, Oleksandr Sotnychenko, Helge Rhodin, Mohammad Shafiei, Hans-Peter Seidel, Weipeng Xu, Dan Casas, and Christian Theobalt. 2017. Vnect: Real-time 3d human pose estimation with a single rgb camera. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 44.

Christos Mousas. 2017. Full-Body Locomotion Reconstruction of Virtual Characters Using a Single Inertial Measurement Unit. *Sensors* 17, 11 (2017), 2589.

Richard A Newcombe, Dieter Fox, and Steven M Seitz. 2015. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 343–352.

Alejandro Newell, Kaiyu Yang, and Jia Deng. 2016. Stacked hourglass networks for human pose estimation. In *ECCV*. 483–499.

Iasonas Oikonomidis, Nikolaos Kyriazis, and Antonis A Argyros. 2012. Tracking the articulated motion of two strongly interacting hands. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 1862–1869.

Mohamed Omran, Christoph Lassner, Gerard Pons-Moll, Peter Gehler, and Bernt Schiele. 2018. Neural Body Fitting: Unifying Deep Learning and Model Based Human Pose and Shape Estimation. In *International Conference on 3D Vision (3DV)*.

Xue Bin Peng, Glen Berseth, Kangkang Yin, and Michiel Van De Panne. 2017. DeepLoco: Dynamic Locomotion Skills Using Hierarchical Deep Reinforcement Learning. *ACM Trans. Graph.* 36, 4, Article 41 (July 2017), 13 pages. https://doi.org/10.1145/3072959.3073602

Gerard Pons-Moll, Andreas Baak, Juergen Gall, Laura Leal-Taixe, Meinard Mueller, Hans-Peter Seidel, and Bodo Rosenhahn. 2011. Outdoor Human Motion Capture using Inverse Kinematics and von Mises-Fisher Sampling. In *IEEE International Conference on Computer Vision (ICCV)*. 1243–1250.

Gerard Pons-Moll, Andreas Baak, Thomas Helten, Meinard Müller, Hans-Peter Seidel, and Bodo Rosenhahn. 2010. Multisensor-Fusion for 3D Full-Body Human Motion Capture. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Gerard Pons-Moll, Sergi Pujades, Sonny Hu, and Michael Black. 2017. ClothCap: Seamless 4D Clothing Capture and Retargeting. *ACM Transactions on Graphics* 36, 4 (2017), 73:1–73:15.

Gerard Pons-Moll, Javier Romero, Naureen Mahmood, and Michael J Black. 2015a. Dyna: A model of dynamic human shape in motion. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 120.

Gerard Pons-Moll, Jonathan Taylor, Jamie Shotton, Aaron Hertzmann, and Andrew Fitzgibbon. 2015b. Metric Regression Forests for Correspondence Estimation. *International Journal of Computer Vision (IJCV)* (2015), 1–13.

Helge Rhodin, Christian Richardt, Dan Casas, Eldar Insafutdinov, Mohammad Shafiei, Hans-Peter Seidel, Bernt Schiele, and Christian Theobalt. 2016. EgoCap: egocentric marker-less motion capture with two fisheye cameras. 35, 6 (2016), 162.

Helge Rhodin, Nadia Robertini, Christian Richardt, Hans-Peter Seidel, and Christian Theobalt. 2015. A versatile scene model with differentiable visibility applied to generative pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*. 765–773.

Daniel Roetenberg, Henk Luinge, and Per Slycke. 2007. Moven: Full 6dof human motion tracking using miniature inertial sensors. *Xsen Technologies, December* (2007).

Nikolaos Sarafianos, Bogdan Boteanu, Bogdan Ionescu, and Ioannis A Kakadiaris. 2016. 3d human pose estimation: A review of the literature and analysis of covariates. *Computer Vision and Image Understanding* 152 (2016), 1–20.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45, 11 (1997), 2673–2681.

L. Schwarz, D. Mateus, and N. Navab. 2009. Discriminative Human Full-Body Pose Estimation from Wearable Inertial Sensor Data. *Modelling the Physiological Human* (2009), 159–172.

Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. 2013. Real-time human pose recognition in parts from single depth images. *Commun. ACM* 56, 1 (2013), 116–124.

L. Sigal, A.O. Balan, and M.J. Black. 2010. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal on Computer Vision (IJCV)* 87, 1 (2010), 4–27.

R. Slyper and J. Hodgins. 2008. Action capture with accelerometers. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '08)*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 193–199. http://dl.acm.org/citation.cfm?id=1632592.1632620

Jonathan Starck and Adrian Hilton. 2003. Model-based multiple view reconstruction of people. In *null*. IEEE, 915.

Carsten Stoll, Nils Hasler, Juergen Gall, Hans-Peter Seidel, and Christian Theobalt. 2011. Fast articulated motion tracking using a sums of gaussians body model. In *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 951–958.

Yu Tao, Zerong Zheng, Kaiwen Guo, Jianhui Zhao, Dai Quionhai, Hao Li, Gerard Pons-Moll, and Yebin Liu. 2018. DoubleFusion: Real-time Capture of Human Performance with Inner Body Shape from a Depth Sensor, In IEEE Conf. on Computer Vision and Pattern Recognition. *IEEE Conf. on Computer Vision and Pattern Recognition*. CVPR Oral.

Jochen Tautges, Arno Zinke, Björn Krüger, Jan Baumann, Andreas Weber, Thomas Helten, Meinard Müller, Hans-Peter Seidel, and Bernd Eberhardt. 2011. Motion reconstruction using sparse accelerometer data. *ACM Transactions on Graphics (TOG)* 30, 3 (2011), 18.

Jonathan Taylor, Jamie Shotton, Toby Sharp, and Andrew Fitzgibbon. 2012. The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 103–110.

Bugra Tekin, Pablo Márquez-Neila, Mathieu Salzmann, and Pascal Fua. 2016. Fusing 2D Uncertainty and 3D Cues for Monocular Body Pose Estimation. *arXiv preprint arXiv:1611.05708* (2016).

Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. 2014. Joint training of a convolutional network and a graphical model for human pose estimation. In *NIPS*. 1799–1807.

Alexander Toshev and Christian Szegedy. 2014. Deeppose: Human pose estimation via deep neural networks. In *CVPR*. 1653–1660.

Matthew Trumble, Andrew Gilbert, Charles Malleson, Adrian Hilton, and John Collomosse. 2017. Total capture: 3d human pose estimation fusing video and inertial sensors. In *Proceedings of 28th British Machine Vision Conference*. 1–13.

Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. 2016. WaveNet: A Generative Model for Raw Audio. *CoRR* abs/1609.03499 (2016). arXiv:1609.03499 http://arxiv.org/abs/1609.03499

Daniel Vlasic, Rolf Adelsberger, Giovanni Vannucci, John Barnwell, Markus Gross, Wojciech Matusik, and Jovan Popović. 2007. Practical motion capture in everyday surroundings. *ACM Transactions on Graphics (TOG)* 26, 3, 35.

Timo von Marcard, Roberto Henschel, Michael Black, Bodo Rosenhahn, and Gerard Pons-Moll. 2018. Recovering Accurate 3D Human Pose in The Wild Using IMUs and a Moving Camera. In *European Conference on Computer Vision (ECCV)*.

Timo von Marcard, Gerard Pons-Moll, and Bodo Rosenhahn. 2016. Human Pose Estimation from Video and IMUs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 38, 8 (aug 2016), 1533–1547.

Timo von Marcard, Bodo Rosenhahn, Michael J Black, and Gerard Pons-Moll. 2017. Sparse inertial poser: Automatic 3D human pose estimation from sparse IMUs. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 349–360.

Jindong Wang, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. 2017. Deep learning for sensor-based activity recognition: A survey. *arXiv preprint arXiv:1707.03502* (2017).

Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. 2016. Convolutional pose machines. In *CVPR*. 4724–4732.

Xiaolin Wei, Peizhao Zhang, and Jinxiang Chai. 2012. Accurate realtime full-body motion capture using a single depth camera. *ACM Transactions on Graphics (TOG)* 31, 6 (2012), 188.

Xiaowei Zhou, Menglong Zhu, Spyridon Leonardos, Konstantinos G Derpanis, and Kostas Daniilidis. 2016. Sparseness meets deepness: 3D human pose estimation from monocular video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4966–4975.

Michael Zollhöfer, Matthias Nießner, Shahram Izadi, Christoph Rehmann, Christopher Zach, Matthew Fisher, Chenglei Wu, Andrew Fitzgibbon, Charles Loop, Christian Theobalt, et al. 2014. Real-time non-rigid reconstruction using an RGB-D camera. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 156.

A ADDITIONAL ARCHITECTURES

Along with the models discussed in the paper, we experimented with other, non-recurrent structures. Specifically, we implemented a WaveNet architecture [van den Oord et al. 2016] and a simple feed-forward network (FFN). The FFN is composed of 5 fully-connected layers with 256, 512, 512, 256, and 256 units per layer respectively. At a single time step t the model is fed a temporal window of 20 past and 5 future frames. Table 4 summarizes the results in terms of mean joint angle error.

The FFN performs around 4.4° worse on TotalCapture than our best BiRNN evaluated on 20 past and 5 future frames. Additionally, the output is greatly corrupted by jerkiness and trembling artifacts. WaveNet performs better both in terms of the joint angle error and visual quality. Although WaveNet is able to considerably reduce the trembling artifacts, they are still apparent, resulting in displeasing visual output.

Furthermore, the BiRNN model proposed in this paper offers much greater flexibility. Increasing the input window size in a feed-forward network requires both retraining the model and incurs a large growth in trainable parameters. This is not the case for the BiRNN; the window length can be changed "on the fly" for the same model and hence does not affect the number of parameters.

Table 4. Performance of WaveNet and a feed-forward network (FFN) on TotalCapture and DIP-IMU in terms of the mean joint angle error in degrees. Both models were trained on the synthetic AMASS training set.

	TotalCapture $\mu(\pm std)$	DIP-IMU $\mu(\pm std)$
WaveNet	17.26 (±13.91)	30.79 (±17.98)
Feed-forward	20.32 (±15.67)	31.88 (±20.63)

B ADDITIONAL FIGURES

Fig. 12 shows the architecture details of the BiRNN as reported in the paper. In Fig. 13 we show the three poses with the highest mean joint angle error taken from the test set of DIP-IMU.

C NORMALIZATION

In Section 4.3 we show how the data is normalized w.r.t. the root. We experimented with more normalization schemes, which we did not find to be beneficial and explain in the following.

Per-sequence normalization. Instead of normalizing the sensor measurements per frame, we experimented with normalizing them only to the root orientation in the initial frame of the sequence. In this case, we feed measurements of all 6 sensors into the model

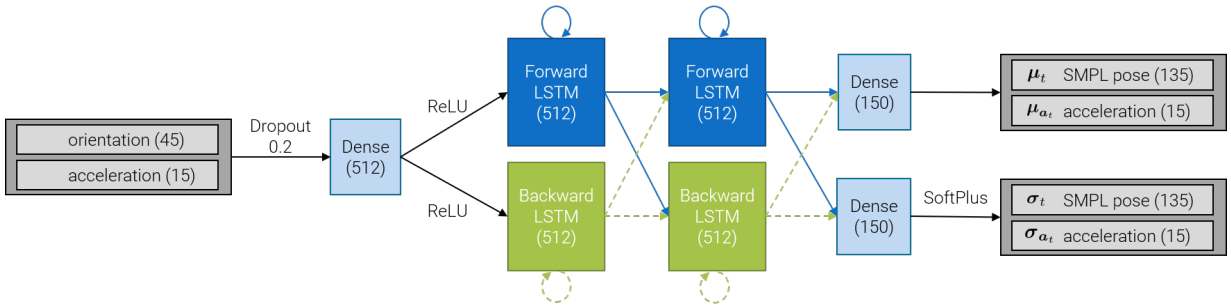

Fig. 12. Network architecture details. Numbers in brackets are input/output dimensions or number of units in the respective layer. From left to right: The normalized accelerations and orientations are passed to a dense layer before being fed into the two bidirectional recurrent layers as outlined in Fig. 3. The output of the last recurrent layer is mapped to two output vectors: the mean SMPL parameters $\boldsymbol{\mu}_t$ and mean accelerations $\boldsymbol{\mu}_{\boldsymbol{a}_t}$ and the respective standard deviations $\boldsymbol{\sigma}_t$ and $\boldsymbol{\sigma}_{\boldsymbol{a}_t}$. The output of the second dense layer is activated using SoftPlus to enforce non-negativity of the standard deviations.

(instead of 5 in the current architecture), whereby the root orientation in the first frame is always the identity. In other words, the normalization is performed as follows for all sensors $s \in \{1, \ldots, 6\}$:

$$\bar{\mathbf{R}}_s^{TB}(t) = \mathbf{R}_{\text{root}}^{-1}(0) \cdot \mathbf{R}_s^{TB}(t),$$
$$\bar{\boldsymbol{a}}_s(t) = \mathbf{R}_{\text{root}}^{-1}(0) \cdot (\boldsymbol{a}_s(t) - \boldsymbol{a}_{\text{root}}(t))$$

Per-frame heading removal. Another option is to normalize only w.r.t. the heading of the root sensor. We implemented this strategy as follows: We extract the yaw angle γ from the root orientation and create a new rotation matrix $\mathbf{R}_{\text{yaw}}$ that rotates around the y-axis by γ. $\mathbf{R}_{\text{yaw}}$ then replaces $\mathbf{R}_{\text{root}}$ in Equations (9) and (10) to perform the normalization for all sensors $s \in \{1, \ldots, 6\}$ and all time steps t. Training our BiRNN with this normalization scheme results in a mean angular error of 40.18° (±21.74°) on TotalCapture, which is 24.41° worse than the per-frame normalization we adopt.

Zero-mean unit-variance. Note that after we normalize orientations and accelerations to the root, we subtract the mean and divide by the standard deviation to rescale the inputs; i.e., the input to the model is $(\mathbf{x}_t - \boldsymbol{\mu})/\boldsymbol{\sigma}$, where the statistics $\boldsymbol{\mu}, \boldsymbol{\sigma} \in \mathbb{R}^{60}$ are computed over the entire training data set. The same normalization procedure is applied to the outputs.

D INFLUENCE OF ACCELERATION

To better understand the influence of acceleration on final pose estimates we conduct a further experiment in which we compare our best network configuration (BiRNN, fine-tuning with accelerations) with a BiRNN configuration that does not use any accelerations in the input. In this experiment, we make two observations. First, the error on both TotalCapture and DIP-IMU is higher (both the mean and the standard deviations). Second, the error on TotalCapture increases to 19.45° after fine-tuning. These findings indicate that both orientations and accelerations are indeed essential to produce accurate predictions, especially for the more challenging poses in DIP-IMU. Furthermore, the higher error on TotalCapture indicates that without accelerations the network overfits to DIP-IMU.

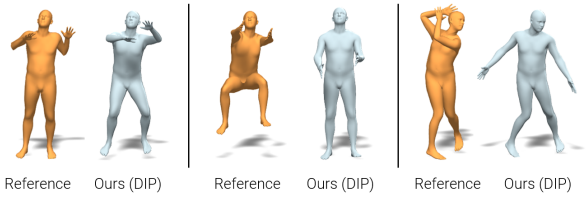

Fig. 13. The three poses with the highest mean joint angle error taken from the test set of DIP-IMU. *Left:* The orientation of the lower arm is reasonable but the model failed to disambiguate the orientation of the upper arms. *Middle:* Same observation but for lower extremities. *Right:* Poor reconstruction of the arms.

Table 5. Performance of a BiRNN that does not use accelerations as inputs (*no acc*) compared to the best model as reported in the paper.

	TotalCapture $\mu(\pm std)$	DIP-IMU $\mu(\pm std)$
BiRNN (fine-tuning)	16.84 (±13.22)	17.54 (±11.54)
BiRNN (fine-tuning, no acc)	19.45 (±15.67)	18.89 (±15.24)

E HARDWARE SPECIFICATIONS

We trained our models on a NVIDIA GTX Titan X (Pascal, 12 GB), which took roughly 3 hours for our best BiRNN model. In the live demo, both the visualization (in Unity) and the model inference (in Python) run on the same machine, i.e., both processes access the Titan X GPU. Because Unity and Python communicate through the network stack, it is possible to run the visualization component on a different, less potent machine. To test this, we run the visualization on a commodity laptop, an Asus Zenbook (CPU i7-3632QM @ 2.20 GHz, 8 GB RAM, NVIDIA GeForce GT 650M (2 GB)).

Table 7 summarizes the resulting FPS for these different settings and for 4 different models. Reported is the time it takes to grab the current measurements from the IMUs, send them to the model over the network and retrieve the model's predictions for that time step; i.e., the FPS displayed by Unity might differ. Furthermore, the actual

Table 6. Dataset capture protocol used to record DIP-IMU.

Categories	Motions (# Repetitions)	# Frames	Minutes
Upper Body	Arm raises, stretches, and swings (10). Arm crossings on torso and behind head (10).	116,817	32.45
Lower Body	Leg raises (10). Squats (shoulder-width and wide) (5). Lunges (5).	70,743	19.65
Locomotion	Walking straight (3). Walking in circle (2). Sidesteps, crossing legs (1). Sidesteps, touching feet (1).	73,935	20.54
Freestyle	The subject can select one of the following activities: jumping jacks, tennis, kicking/boxing, push-ups, basketball. Choice of jumping jacks is predominant.	18,587	5.16
Interaction	The subject sits at a table and interacts with everyday objects, such as keyboard, mobile device, toys, grabbing objects in front of them, touching mounted screen displays. Freestyle for 1 minute.	50,096	13.92

runtime of the RNN model is technically capped by the update rate of the Xsens sensors (60 Hz).

F DATA COLLECTION

Table 6 summarizes the protocol we used for collecting our new *real* dataset, DIP-IMU. Particular attention was paid to recording activities that are underrepresented in existing Mocap datasets.

Table 7. Average running time in FPS for the model inference component of our live demo. Measured is the time it takes to grab the current measurements from the IMUs, send them to the model over the network and retrieve the predictions for that time step. *Local* means visualization and model inference run on the same machine (here the machine used for training). *Remote Laptop* means the visualization runs on a commodity laptop.

	Local [fps]	Remote Laptop [fps]
RNN	173.3	67.3
BiRNN (20, 5)	29.7	25.1
BiRNN (50, 5)	15.7	14.8
BiRNN (100, 5)	9.3	8.7