Towards Single Camera Human 3D-Kinematics

Bittner, M.; Yang, W.; Zhang, X.; Seth, A.; van Gemert, J.C.; van der Helm, F.C.T.

DOI
10.3390/s23010341

Publication date
2022

Document Version
Final published version

Published in
Sensors

sensors

MDPI

Article

Towards Single Camera Human 3D-Kinematics

Marian Bittner [1,2,3,*,†], Wei-Tse Yang [2,†], Xucong Zhang [2], Ajay Seth [3], Jan van Gemert [2] and Frans C. T. van der Helm [3]

[1] Vicarious Perception Technologies (VicarVision), 1015 AH Amsterdam, The Netherlands
[2] Computer Vision Lab, Delft University of Technology, 2628 XE Delft, The Netherlands
[3] Biomechanical Engineering, Delft University of Technology, 2628 CN Delft, The Netherlands
* Correspondence: mbittner.work@gmail.com
† These authors contributed equally to this work.

Abstract: Markerless estimation of 3D Kinematics has the great potential to clinically diagnose and monitor movement disorders without referrals to expensive motion capture labs; however, current approaches are limited by performing multiple de-coupled steps to estimate the kinematics of a person from videos. Most current techniques work in a multi-step approach by first detecting the pose of the body and then fitting a musculoskeletal model to the data for accurate kinematic estimation. Errors in training data of the pose detection algorithms, model scaling, as well the requirement of multiple cameras limit the use of these techniques in a clinical setting. Our goal is to pave the way toward fast, easily applicable and accurate 3D kinematic estimation . To this end, we propose a novel approach for direct 3D human kinematic estimation D3KE from videos using deep neural networks. Our experiments demonstrate that the proposed end-to-end training is robust and outperforms 2D and 3D markerless motion capture based kinematic estimation pipelines in terms of joint angles error by a large margin (35% from 5.44 to 3.54 degrees). We show that D3KE is superior to the multi-step approach and can run at video framerate speeds. This technology shows the potential for clinical analysis from mobile devices in the future.

Keywords: 3D-kinematics; 3D-kinematic estimation; OpenSim; pose estimation; musculoskeletal modelling; markerless motioncapture

Citation: Bittner, M.; Yang, W.; Zhang, X., Seth, A.; van Gemert, J; van der Helm, F.C.T. Towards Single Camera Human 3D-Kinematics . *Sensors* **2023**, *23*, 341. https://doi.org/10.3390/s23010341

Academic Editors: Francesco E. Pontieri and Catherine Disselhorst-Klug

Received: 1 November 2022
Revised: 17 December 2022
Accepted: 21 December 2022
Published: 28 December 2022

1. Introduction

3D Human kinematics involves measuring joint angles between body segments, which is essential in the day-to-day practice of experts. Skilled physicians could judge, just by looking at a specific motion of their patient, whether it is healthy or abnormal. Skilled sports coaches can help their coachees achieve better performance and lower injury risk by evaluating their movements through observation. However, these visual examinations of human kinematics remain inherently subjective, leading to variation between and within human observers. Modern systems and sensors could reduce these variations through more objective observations. Yet, these systems make the measurement of human motion more costly and more time-consuming. A system with the availability and ease of use of visual estimation would help physicians and coaches make more objective observations more often, ultimately raising their own and their subjects quality of life. Digital cameras have made the estimation of human kinematics more accessible but come at the cost of reduced accuracy. Compared to the more traditional Optical Motion capture (OMC) systems, markerless motion capture (MMC) systems do not require specialized cameras and markers attached to the subject being monitored, but use normal RGB cameras in combination with image-based automatic pose estimation algorithms. Instead of specific markers, pose estimation algorithms detect the centers of major joints of the human body, such as the shoulders, hips, and knees. These detected centers are usually referred to as key points.

Multiple commonly used markerless motion capture methods rely on 2D pose estimation methods [1–4]. Often these methods still need more than one camera to generate a good estimation of the keypoints in 3D, which again requires additional cameras to be set up. On the other hand, an increasing number of methods are using single-view (monocular) 3D pose estimation methods [5–7], which allow to estimate a 3D pose just by using a single camera. This makes MMC systems faster and more accessible as they do not require the additional time and expertise to place markers on the subject or calibrate multiple cameras. However, MMC systems assume that current pose estimation algorithms can accurately replace markerless motion capture systems for, e.g., biomechanical applications [8–10].

Commonly used pose estimation algorithms introduce mistakes in kinematic estimation pipelines due to systematic errors in their predictions. To detect key points, most pose estimation methods are trained on a combination of images of a person and ground truth annotations which map pixels in the image to their corresponding joint center.These ground truth annotations are often manually conducted by non-expert annotators, leading to errors caused by personal biases for training and inaccuracies in the pose estimations [9]. For example, Needham et al. [11] compared three often used pose estimation algorithms OpenPose [12], DeepLabCut [13] and AlphaPose [14] algorithm against an OMC system and showed errors in the estimation of joint centers of 30 mm to 50 mm with variations in 12 mm to 25 mm in marker placement. Cronin [9] provides an overview of additional problems with 2D pose estimation for kinematic analysis. These differences are most likely due to a difference between the application that pose estimation algorithms are often developed for and their application to, e.g., the biomedical domain, which has different accuracy requirements [8]. Wade et al. [10] proposed to solve this problem by re-annotating existing large-scale datasets, this, however, is a time-consuming process, when for example considering the COCO-keypoint dataset https://cocodataset.org/#keypoints-2020 (accessed on 2 December 2022) consists of more than 250.000 labeled poses. For the evaluation of pose estimation algorithms, these labeling errors will just appear as a baseline error that all algorithms training on the same data will have. However, for applications in the biomedical domain and in situations such as kinematic estimation, where the pose is just an intermediate step errors can propagate to subsequent tasks.

Errors in the estimated pose cannot not be corrected by most kinematic estimation pipelines because they all roughly follow a 'multi-step' approach. The 'multi-step' approach consists of

- Detection of the 3D pose (in one or more steps);
- (Optional) modeling of the pose with a (musculo)skeletal model.
- Calculation of kinematics and/or downstream tasks such as gait parameters or dynamics.

For example, Kidzinski et al. [1] used OpenPose to first predict key points from a video and then trained a convolutional neural network (CNN) to predict the walking parameters of patients with cerebral palsy. Liao et al. [6] first model the 2D pose in OpenPose then create a 3D pose using data-driven matching and finally estimate 3D gait parameters. Noteboom et al. [7] first used VideoPose3D [15] to estimate a 3D pose, followed by modeling in OpenSim [16] for the estimation of dynamics from a single camera. Pagnon et al. [2,3] developed the handy Pose2Sim tool, which first combines 2D OpenPose pose estimations from multiple cameras into a 3D pose then models it in OpenSim. Because the pose estimation step is de-coupled from kinematic estimation, errors in pose estimation propagate through to the estimation of kinematics. Uchida and Seth [17] showed that 20 mm of marker uncertainty leads to a variation of 15.9° in peak ankle plantarflexion angle and impacts downstream tasks such as joint moment estimation. Della Croce et al. [18] showed precision variation 13 mm to 25 mm, which leads to differences in estimated joint angles up to 10°. Fonseca et al. [19] showed that misplacement of markers up to 10 mm can lead to errors of 7° depending on the marker. With estimation errors of 30 mm to 50 mm in keypoint estimation [11], it is to be expected that these errors will substantially influence kinematic estimation from markerless motion capture. Low-pass filter [2,20] or bi-directional Kalman-filter [20] has been applied to compensate for noisy key point

estimations, but cannot correct for faults in keypoint detection. Subsequent modeling and kinematic calculation steps can only compensate for these inaccuracies. This 'multi-step' approach is probably inspired by the steps of a traditional OMC method, as in the traditional OMC systems the pose detection step is done using a different system and is thus isolated from the other steps. In camera-based kinematic estimation pipelines, however, the de-coupling of individual steps is no longer necessary.

Deep neural networks have often demonstrated their ability to outperform multi-step systems, by implicitly learning individual steps through end-to-end training between an input and the desired output [21,22]. The main strength of deep neural networks lies in their ability to break down a highly complex task, in this case, the estimation of kinematics from videos, into a sequence of simpler tasks, without the need for intermediate 'hand-crafted' representations [23,24]. Due to the fully differentiable nature of neural networks, it means that an error in estimation during training can influence all stages of the network and adjust them accordingly [24]. This allows deep neural networks to directly estimate kinematics.

In this work, we challenge the notion of the classical multi-step approach of pose estimation, fitting of a musculoskeletal model and kinematic estimation. To this end, we propose a novel end-to-end method that allows for direct estimation of human kinematics, which is directly optimized for kinematic estimation while treating pose estimation only as an auxiliary task to constrain the estimations of the network. Figure 1 shows a general overview of our method.

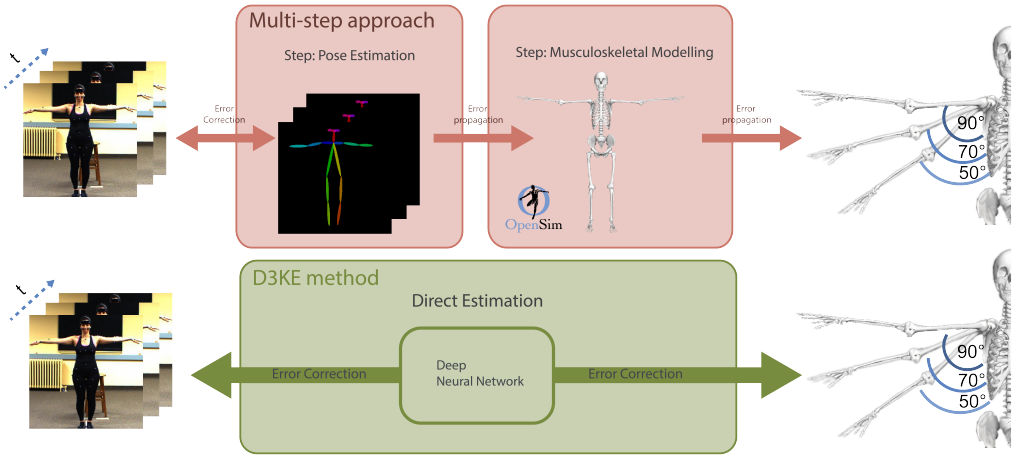

Figure 1. Overview of the proposed direct 3D human kinematics estimation (D3KE). Instead of using the common 'multi-step' approach of predicting pose, fitting it to a model, and estimating kinematics, our D3KE directly estimates the kinematics. Errors in earlier steps of the multi-step approach propagate to later steps; in contrast, our method can correct for errors occurring anywhere between input and output.

Contributions

To the best of our knowledge, we are the first to present an end-to-end trainable network that directly generates joint angles, joint positions, scale factors and marker positions of a biomechanical model from a monocular video. We propose a method that directly regresses from a video to joint angles and scales using deep neural networks. We investigate the influence of various temporal smoothing methods to increase the accuracy of our algorithm. We introduce a novel type of network layer that allows for the calculation of the 3D pose from estimated kinematics during the training process to train the network simultaneously on the pose and kinematic labels.

2. Materials and Methods

Our method takes videos from a single camera as input and directly estimates joint angles, which we call direct 3D kinematic estimation (D3KE). The proposed method first coarsely estimates kinematics per frame by using a convolutional neural network, and then it uses a sequence network with temporal relations across frames to re-fine kinematic estimations at each frame. An overview of our method is shown in Figure 2. Both networks estimate the scale of body segments, joint angles, and a rotation matrix from the pelvis to the ground, those serve as input for a skeletal-model layer in both networks that allows for additional supervision on the pose of a subject.

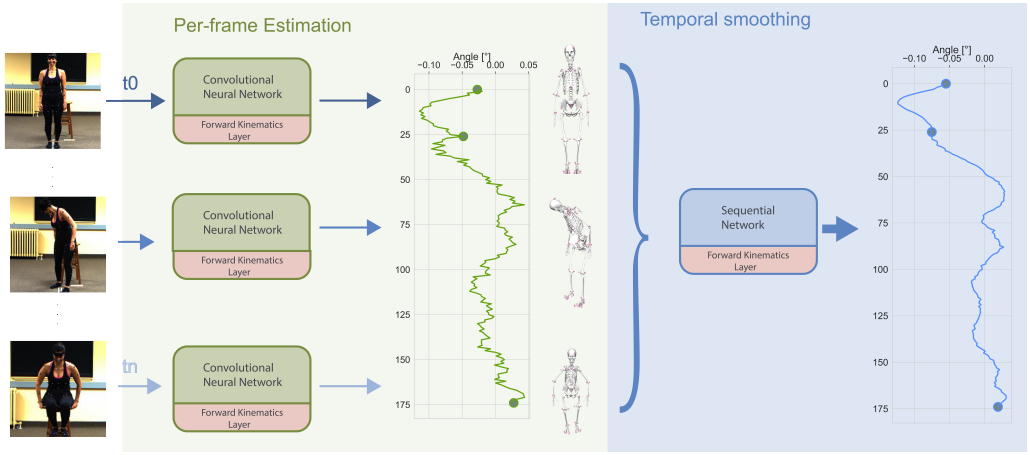

Figure 2. Taking a single view video as input, D3KE consists of one convolutional neural network and one sequential network. Per frame, D3KE outputs joint angles and scales of individual bones in a skeletal model(scale factors) with a convolutional network. Additionally, joint angle and scale factor are converted to a pose through the skeletal-model kinematics (SM) layer. A series of frame estimations in time are then fed into a sequential network to smooth the estimations and reduce artifacts if one limb occludes another in the view of the camera (self-occlusion).

In this section, we first describe the deep learning architecture, including a detailed description of the skeletal-model layer. We then describe how the ground truth data was generated and which pre-processing and hyperparameters were used for training. Lastly, we describe the dataset used for training and testing our method.

2.1. Network Structure

Convolutional neural networks(CNNs) have shown good accuracy for 2D and 3D pose estimations [12,25–28] from single input images. Conventionally 2D CNNs are used for pose estimation tasks, that takes a single image as an input and predict the pose of one or multiple people in the image. For our method, we choose a per-frame convolutional network to coarsely predict the joint angle and scaling parameters. Inspired by [25], we choose a standard pre-trained ResNeXt-50 [29] as our convolutional backbone.

To fine-tune the per-frame predicted joint angles and scaling parameters we add a sequential network. Sequential networks are used in pose estimation to 'lift' an estimated 2D pose to 3D [30,31]. Recent research combines temporal information with lifting to improve accuracy during frames where one limb occludes another in the view of the camera (self-occlusion) or where not all key points were detected [15,32,33]. In contrast to CNNs, these sequential networks do not take a single frame as input but exploit temporal dependencies in the data for their prediction. As the convolutional network outputs per-frame estimates, it cannot take temporal information into account. We add a sequential network to our architecture to refine a sequence of estimations made by the convolutional model. Inspired by works on temporal lifting we experimentally evaluate three sequential networks; an LSTM [34], a Temporal Convolutional Network (TCN) [15] and a Transformer [33] to refine the predicted joint angles and scale factors.

Both the convolutional and the sequential network contain a specialized layer that allows each network to perform the kinematic transformations of a musculo-skeletal model. Therefore, at train time both networks can be supervised not only on the estimated joint angles but also on a resulting pose.

Both convolutional and sequential networks are supervised by losses of joint positions, marker positions, body scales and joint angles. The overall objective function L can be expressed in the equation

$$L = \lambda_1 L_{joint} + \lambda_2 L_{marker} + \lambda_3 L_{body} + \lambda_4 L_{angle}, \quad (1)$$

where $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are weights of losses. We use the root-relative L1 loss in Equation (2) to define the loss of marker position L_{marker} and the loss of joint position L_{joint}. The estimations $\hat{y}$ and the labels y are first subtracted with each root position $\hat{y}_{root}, y_{root}$. For the loss of body scales L_{body} and joint angles L_{angle}, we calculate the L1 norm.

$$l = \|(\hat{y} - \hat{y}_{root}) - (y - y_{root})\|_1 \quad (2)$$

The objective of a neural network during training is to minimize the loss function; in our case, the difference between estimated and ground truth joint angles. However, this joint angle loss cannot capture the underlying relations and constraints of individual angles, dictated by the human musculoskeletal system. Intuitively, small changes in the angles of spine, shoulder, and elbow can accumulate and lead to large differences in the position of the hand, as illustrated in Figure 3. To address this issue, we propose to use a skeletal-model layer to perform the kinematic transform of a musculoskeletal model.

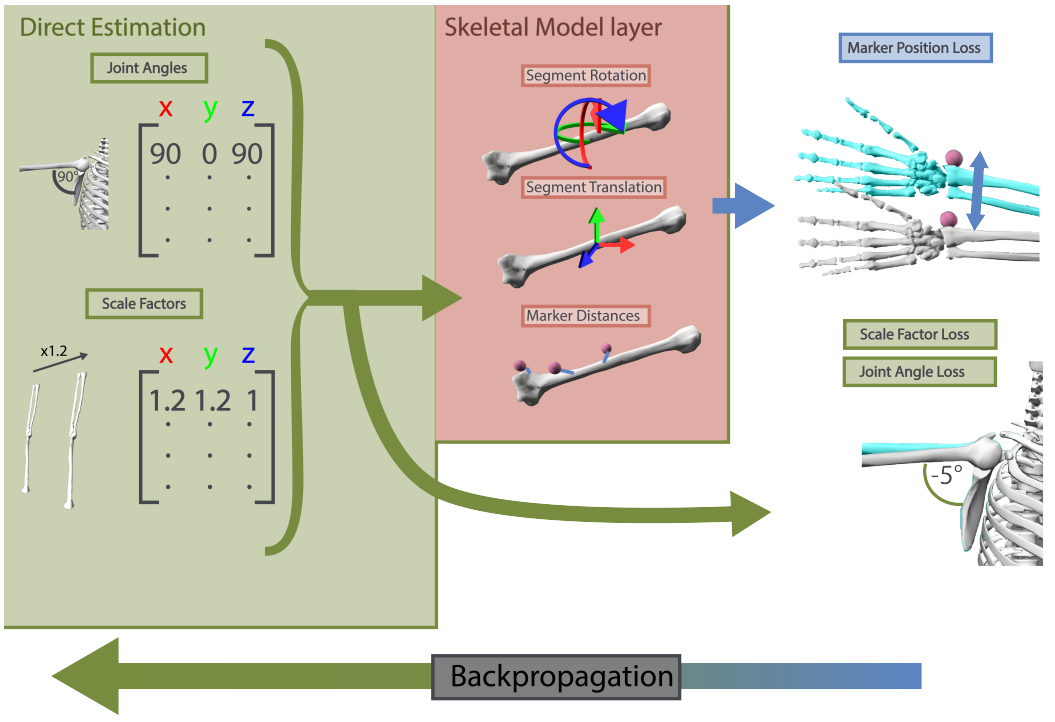

Figure 3. Our skeletal-model layer uses an internal representation of a skeletal model to convert the predicted joint angles and scale factors to the positions of individual markers on segments of the skeletal model. This allows our method to be supervised during training not only on errors(losses) in the estimation of joint angles but also on errors in the resulting pose. On the right, we show the additional error that is created between estimations (gray) and ground truth (blue). This auxiliary estimation of the pose as 3D marker positions helps to constrain the estimation of joint angles as small changes in proximal joints can have a large effect on a marker at more distal positions.

Skeletal-Model Layer

The skeletal-model layer allows us to convert predicted joint angles into marker positions on a skeletal model and add them as an additional loss term. This loss term represents the cumulative effect of small joint angle changes on the final pose, indirectly imposing the constraints of a skeletal-model on the predictions of the network. As the skeletal-model layer does not contain any learnable parameters, i.e., it cannot change during network training. The accuracy of the predicted pose is completely determined by the input to the skeletal-model layer; thus, the pose prediction is only an auxiliary task.

A skeletal model consists of body segments, motions between different body segments (joints) and points with a vector from a center of its anchor body segment (markers). Given body scales β, joint angles θ and rotation matrix $R_{ground \leftarrow pelvis}$, we use the skeletal-model layer to calculate marker positions and joint positions. In the following variables with a hat ($\hat{x}$) denote estimated values, variables without the hat (x) denote the predefined variable from the musculoskeletal model.

First, the translation part T in the transformation from the joint to the body depends on the subject's body scale. For example if the subject has longer legs, the center of the femur will be farther from the hip joint. We can update the translation part $\hat{T}$ by comparing the ratio between predicted body scales $\hat{\beta}$ and default body scales β in Equation (3), where $\odot$ is elementwise multiplication, and $\oslash$ is elementwise division.

$$\hat{T} = T \odot (\hat{\beta} \oslash \beta) \tag{3}$$

Then, we create a matrix to represent spatial transformation of motions R_motion using Equation (4) A_1, A_2, A_3 are the predefined axes $\hat{\theta}_1, \hat{\theta}_2, \hat{\theta}_3$ and predicted angels per degree of freedom per joint in axis-angle notation. $G(A, \theta)$ is the standard function converting an axis-angle representation to a 3x3 transformation matrix.

$$R_{motion} = \begin{bmatrix} R_3 R_2 R_1 & 0 \\ 0 & 1 \end{bmatrix}_{4\times4} \tag{4}$$

$$R_1 = G(A_1, \theta_1) \tag{5}$$

$$R_2 = G(R_1 A_2, \theta_2) \tag{6}$$

$$R_3 = G(R_2 R_1 A_3, \theta_3) \tag{7}$$

Then, we can calculate the estimated transformation from the body to its parent body $\hat{R}_{parent \leftarrow child}$ in Equation (9) using Equation (8) with O_{parent}, O_{child} denoting predefined orientations from and $\hat{T}_{parent}, \hat{T}_{child}$ the predicted translations from the joint to the parent/child. $F(O_a)$ the conversion from euler angles to a 3×3 Rotation matrix.

$$R_{parent/child \leftarrow joint}(O_{parent/child}, T_{parent/child}) = \begin{bmatrix} F(O_{parent/child}) & T_{parent/child} \\ 0 & 1 \end{bmatrix}_{4\times4} \tag{8}$$

$$R_{parent \leftarrow child} = R_{parent \leftarrow joint}\, R_{motion}\, R^{-1}_{child \leftarrow joint} \tag{9}$$

We measure the spatial transform by traversing from the root (pelvis) to leaf nodes (hands and feet) in the level order. In D3KE, we directly infer the rotation matrix from the pelvis to the ground $R_{ground \leftarrow pelvis}$. $R_{ground \leftarrow pelvis}$ can initially be expressed in Equation (10), where I denotes the identity matrix. The rotation part of $R_{child \leftarrow joint}$ is also a 3×3 identity matrix in our musculoskeletal model. Our method aims to predict the root-relative position so the translation part can be ignored during the prediction. Moreover, in our musculoskeletal model, only joint angles of the pelvis are unbounded in $[-\infty, \infty]$. Predicting three

unbounded angles to form the rotation matrix in Equation (4) will have the problem of discontinuity [35]. Thus, we directly predict the rotation matrix $R_{ground \leftarrow pelvis}$.

$$R_{ground \leftarrow pelvis} = I_{4\times4}\, R_{motion}\, R^{-1}_{pelvis \leftarrow joint} \tag{10}$$

Last, a marker with a vector of $\vec{d}$ from the center of the body is also dependent on the body scales. The predicted vector of $\hat{d}$ is updated in Equation (11). The position of the predicted point is calculated in Equation (12) with $\hat{R}_{parent \leftarrow child}$ and $\hat{d}$.

$$\hat{d} = \vec{d} \odot (\hat{\beta} \oslash \beta) \tag{11}$$

$$P = \prod_{parent,child \in path} R_{parent \leftarrow child} \begin{bmatrix} \vec{d} \\ 1 \end{bmatrix}_{4x1} \tag{12}$$

2.2. Network Training

2.2.1. Ground Truth Generation

For training our method, we need to create custom ground truth data that contains all outcomes that our network is predicting since they are not available in publicly available datasets. Most pose estimation datasets, provide only video and marker positions from optical motion capture (OMC) system. For training our method, we need the joint angle and the scales of individual bones, a rotation matrix of the pelvis to the ground as well as the marker positions corresponding to them. To generate these, we model the OMC data, represented as a 3D human mesh model in the OpenSim software [16] and use inverse kinematics [36,37] to generate joint angles. The following describes each step in more detail.

First, we create a general (musculo)skeletal model to fit the data using the OpenSim software [16,38]. As we are interested in capturing the complete motion of the human subject, we model the full body. With the OpenSim software [16,38] we create a full-body musculoskeletal model (MSM) by merging existing models of upper limbs and lower limbs [39–43] and thoracolumbar spine [44–46]. We add wrist and hand [47,48] models to the MSM, which are not used for ground truth generation, for the sake of aesthetics. The full-body model contains all bones in a skeletal system from the head to feet and from the upper arms to the hands. We do not model every degree of freedom between vertebrae to avoid expensive computation and the requirement of at least three markers to measure the motions of one vertebra. Instead, we separate the spine from the fifth lumbar to the first cervical vertebra into nine segments.

Then, we fit our data to the musculoskeletal model. Instead of using the OMC marker data directly, we use OMC marker converted to 3D human mesh representations using the MoSh++ [49] method, to make scaling the model to individual participants more time efficient and allow us to define an arbitrary number of virtual markers. We fit our data to the musculoskeletal model, by first defining virtual markers on the vertices of the 3D mesh representation. We then used these virtual markers as input for the OpenSim software. Then, we used the OpenSim internal scaling tool to scale the proportion of individual body segments according to the distances of virtual markers on the 3D mesh. As the sizes of individual body parts vary across individuals, this step must be conducted individually for each subject in the dataset. We define the ratio in dimensions between the default and scaled body segments as scaling factors. Finally, we used the inverse kinematics solver for the calculation of joint angles. During this process, the MSM is moved for each time step to a position that minimizes the sum of weighted squared errors between the virtual markers on the 3D mesh and markers defined on the musculoskeletal model. All joint angles where segments had a higher squared error than 2 cm were disregarded in the analysis.

The final ground truth values were the calculated joint angles, the scaling factors per segment as well as the virtual marker positions. Additionally, a pelvis rotation matrix was

generated for each frame, since the pelvis functions as the relative position of the model to the ground that is free to move in all directions.

2.2.2. Data Preparation and Hyperparameters

To generate the input for our network, each video frame was cropped and augmented. We use the pre-trained Faster R-CNN [50] with ResNet-50 [51] backbone to extract a square bounding box of the person in videos and resize it to 256 × 256 pixels as the input image size. During training, we apply data augmentation with scaling, rotation, translation and noise to simulate occlusions similar to [25].

Our model was trained using the following hyperparameters and loss. For the ResNeXt model, we use an Adam optimizer with weight decay [52] of 0.001 and a batch size of 64. The learning rate exponentially decays in two steps from 5×10^{-4} to 3.33×10^{-5} over 28 epochs and from 3.33×10^{-6} to 10^{-6} over 2 epochs. For both sequential and convolutional networks, we set the hyperparameters with $\lambda_1 = 1.0$, $\lambda_2 = 2.0$, $\lambda_3 = 0.1$ and $\lambda_4 = 0.06$ experimentally. Due to memory constraints, we do not train convolutional and sequential models simultaneously, but in succession, by first training the convolutional model and then refining predictions using the sequential model.

2.3. Software Tools

All training was conducted in python using the PyTorch library [53]. The pre-trained ResNext and FasterRCNN networks were obtained from the torchvision library [54]. All code for training and generation of ground truth will be made available in a Github repository: https://github.com/bittnerma/Direct3DKinematicEstimation (accessed on 30 September 2022).

2.4. Data

We trained and tested D3KE on the BML-MoVi Database [55]. BML-Movi is an extensive motion capture and video dataset, it contains recordings of 90 actors that each perform 20 kinds of everyday movements as well as a random one. Motions were captured using inertial measurement units as well as a Qualisys optical motion capture system and videos were recorded using two computer-vision cameras. For this study, we used recordings from the calibrated Point Gray cameras (PG1, PG2) during recording session F as the full set of optical markers was used during this session. In accordance with the anatomical plane that each camera is viewing during the initial T-Pose of the participants, we will refer to the camera view captured by PG1 as the frontal- and PG2 as the sagittal camera view. For the generation of ground truth virtual markers, we use the 3D mesh representations of the Qualisys data that is provided in the larger AMASS dataset [56]. For analysis of the data, we divided the BML-Movi database into 63 participants for training, 16 participants for the testing, and three participants for validation. This is common practice in the supervised training of deep neural networks [57]. At training time, the validation set is used to evaluate the accuracy of kinematic estimation after each training iteration on a portion of the data the network does not have access to, to prevent overfitting on the training set.

3. Experiments

3.1. Experiment 1: Direct vs. Multi-Step Estimation

To evaluate the accuracy of our direct 3D kinematic estimation approach (D3KE) for joint angle estimation, we compare its performance against multiple versions of the multi-step approach.

3.1.1. Experiment 1-A: 3D Pose Based Kinematic Estimation

We first compare our direct estimation of kinematics and a 3D pose estimation multi-step baseline. To create a fair comparison between direct and multi-step estimations, we implement a custom multi-step approach (CMS) that is trained on the same data as

our direct approach. For the CMS, we combine a *3D* human pose estimation method with subsequent musculoskeletal modeling in OpenSim. We modify the metric-scale heatmaps [25] of the convolutional network to predict marker positions and SMPL keypoint positions in the metric scale. As for D3KE, we exploit a sequence network to re-fine marker positions at the target frame. More specifically, the convolutional network initially infers marker positions under a calibration pose (T-pose), and OpenSim utilizes the predicted marker data for body scaling, where the general musculoskeletal model is scaled to the participant's body size. Re-fined marker positions are then used to run inverse kinematics with the scaled musculoskeletal model to obtain joint angles. The main difference between the CMS approach and D3KE is that CMS uses multiple steps to estimate the kinematics and is only supervised on the marker/pose estimation task, while D3KE is directly trained on the kinematic estimation task; this way, we can compare direct vs. multi-step estimation of kinematics. We use multiple metrics for the comparison of D3KE and the CMS. The mean per bony landmarks position error (MPBLPE) is used to evaluate bony landmark positions. Bony landmarks are markers placed where bones are close to the surface, such as the elbow. This metric is inspired by the mean per joint position error (MPJPE) which is often used in 3D pose estimation. MPBLPE first aligns estimations and ground truth at the root position and calculates the average Euclidean distance. We directly evaluate the body scale factors by the root mean square error RMSE_{body} on the scalars predicted by the network. However, to present the results in a more intuitive format, we choose the axis along the longest dimension in each body scale and convert the scale of the axis into millimeters and calculate the mean absolute error (MAE_{body}).

3.1.2. Experiment 1-B: 2D-Pose Based Kinematic Estimation

In the previous experiment, we evaluate the multi-step baseline with the *3D* body pose estimation method. However, the use of fully trained *2D* pose estimation algorithms is common in kinematic estimation works [2,3,20]. Therefore, we conduct experiments to compare our method and these 2D-based kinematic estimation methods. In contrast to our CMS method which estimates 3D pose from a single camera estimation, 2D pose estimation methods require at least 2 calibrated cameras for the estimation of 3D keypoints. The use of an additional camera to generate the pose could be an advantage, which the CMS method does not have. We chose a naive implementation of the OpenPose algorithm [12,58], which has extensively been used in related work [2,3,20]. Additionally, we test the MediaPipe implementation of the blazepose algorithm [59], as a more modern 2D algorithm. MediaPipe is easy to use since it is available as a python library, however, in contrast to OpenPose it runs faster, allows for additional smoothing of its predictions, provides more key points, and is labeled on different keypoint labels.

For the OpenPose and MediaPipe, we project the key points to 3D using the BML-Movi camera parameters https://github.com/saeed1262/MoVi-Toolbox (accessed on 2 August 2022) . For OpenPose, we connected missing points (due to self-occlusion) using linear interpolation. For MediaPipe, we chose the highest model complexity (2) and set enabled smooth landmarks for continuous frames of a video. For modeling and inverse kinematics, we follow the same steps as for the CMS method only redefining the positions of markers on the OpenSim model to fit the provided key points.

We compare against an average across both camera views for CMS and D3KE, as MediaPipe and OpenPose need at least two cameras to work. We evaluate performance based on mean absolute error $MAE_{angle}(^\circ)$, the standard deviation of errors $SD_{angle}(^\circ)$ and smoothness of the predictions as the mean velocity of the angle MV_{angle} ($^\circ$/s). The mean velocity error is calculated by the derivative of the landmark position and joint angle data with respect to time.

$$MV_{angle} = \sum_{t=0}^{n} \frac{\frac{|s_t - s_{t+1}|}{\Delta t}}{n} \tag{13}$$

with s_t an individual marker position at time t, Δt is the amount of time between time steps and n is the total number of timesteps.

3.2. Experiment 2: Sequential Network Variants

Since we have multiple options for the sequential networks, we evaluate three to determine whether the additional modeling of temporal dependencies in the data improves the accuracy of our method or not. For subsequent smoothing and reduction of self-occlusion artifacts of the estimations, we test three different networks including LSTM [34], temporal convolutional networks (TCNs) [15], and a lifting Transformer [33] as the sequential network. As smoothing is known to improve the accuracy of multi-step approaches [20], we also evaluate combinations of our CMS model with these sequential networks.

For the LSTM, we implement a bidirectional architecture with a hidden size of 128, three recurrent layers and a dropout probability of 0.1. For TCNs, we follow [15] to exploit 243 frames as the receptive field and make the momentum of batch normalization decay from 0.1 to 0.001. For the lifting Transformer, we use a hidden size of 256 and 8 parallel attention heads in the self-attention layer and a channel size of 512 in the convolutional layer. Each sequential network is trained with a sequence length of 243 frames and a batch size of 128 over 50 epochs with Adam optimizer [60]. The learning rate exponentially decays from 10^{-3} to 5×10^{-6}.

We use the same metrics for the comparison of individual network variants as we used for the comparison of D3KE and CMS. In addition, we investigate the smoothness of the predicted sequences, we estimate the mean velocity (MV) on bony landmark positions and joint angles, denoted as MV_{BL} and MV_{angle}.

3.3. Experiment 3: Processing Speed

One important property of our proposed method for clinical applications is its processing speed. As applications for camera-based kinematic estimation should form an alternative to visual examinations in the future, it should ideally be able to run fast enough to estimate kinematics from video frames as fast as they are collected by a camera, mostly between 15 and 30 frames per second.

We compare the running time on Windows 10 with four core CPU, 52 GB RAM and NVIDIA T4 GPU. We compare the CMS and D3KE method as they both use the same type of convolutional network. We choose the lifting Transformer as the sequential architecture in both the CMS and D3KE. For the CMS method, OpenSim is executed in parallel with four cores. Our report results in frames per second.

3.4. Experiment 4: Generalization Performance

The goal of camera-based kinematic estimation is ultimately to create tools for researchers and clinicians to analyze and diagnose human movement, these tools should not discriminate between different subjects and movements. We analyze whether our method generalizes to different subjects, movements, and joints.

To assess how well D3KE generalizes, we compare the estimates of the proposed method to the ground truth on the time series of each of the 16 participants in the test set with respect to the performed movement, the joint, the camera view and the individual participant. For this test, we use the best performing model from experiment 1 with the lifting transformer.

For all time series of joint angles, mean absolute error (MAE) and Pearson's correlation coefficient (ρ) were calculated between the estimation from D3KE and the ground truth.

Central tendencies in the data are reported as a median and interquartile range of MAE, RMSE and ρ, as the data are not normally distributed, as assessed through visual inspection and confirmed by the Shapiro-Wilk test. For completion, mean and standard deviation are also reported. The absolute values of ρ were categorized as weak, moderate, strong and excellent for $\rho \leq 0.35$, $0.35 < \rho \leq 0.67$, $0.67 < \rho \leq 0.90$ and $0.90 < \rho$, respectively [61].

3.5. Software and Tools

All data analysis was conducted in python 3 [62] using the pandas library to generate descriptive statistics, SciPy library for the calculation of MAE, RMSE and Pingouin library for the calculation of ρ.

4. Results

4.1. Direct vs. Multi-Step Estimation

4.1.1. A: 3D Pose Based Kinematic Estimation

As shown in Table 1, D3KE has better performance than CMS methods in terms of joint angles and body scales, and these two factors are the key to kinematic estimation. Significantly, D3KE reduces 37.4% of errors on MAE_{angle} when comparing the CMS method with the Transformer architecture. Although the proposed method has a slightly larger MPBLPE than the CMS, this metric is not related to the kinematic estimation and is only used as one of the losses during training in the proposed method (e.g., MPBLBE is not minimized). This indicates a gain in accuracy when directly estimating kinematics from the video instead of the multi-step approach of first estimating pose and then estimating kinematics.

Table 1. Comparison of bony landmarks position (MPBLPE), body scales (MAEbody) and joint angles (MAEangle) between the estimation of and the ground truth across all participants, movements, joints and camera views. For the custom multi-step approach (CMS) as well as our proposed method, we compare convolutional networks with different temporal networks. All versions of the proposed method show superior performance for the prediction of body scales and joint angle estimation. All CMSs show superior performance in estimating marker positions. Each method group shows better performance for the task it was optimized for, highlighting the importance of direct optimization. Bold numbers indicate the best performance.

		MPBLPE (mm)	**MAE_{body} (mm)**	**MAE_{angle} (°)**
D3KE	Convolutional	37.78	6.07	3.58
	Conv. + LSTM	37.61	5.97	3.57
	Conv. + TCNs	38.06	5.93	3.54
	Conv. + Transformer	**36.98**	**5.90**	**3.54**
CMS	Convolutional	35.04	6.25	5.89
	Conv.+ LSTM	**33.74**	-	5.79
	Conv.+ TCNs	34.52	-	5.82
	Conv.+ Transformer	34.00	-	**5.66**

In Table 2, we list RMSE and MAE for body scales of selected segments. The results show that the CMS performs better than D3KE on lower limbs, and D3KE performs better than the CMS on upper limbs. The CMS and the proposed method have comparable performance in scale estimation of the pelvis and lower limbs.

Table 2. Errors in scaling factors of the proposed method and the baseline compared against the ground truth. D3KE shows better performance for the upper extremities and slightly worse performance for the lower extremities.

	$RMSE_{body}$ (MAE_{body} (mm))	
	CMS	**D3KE**
pelvis	0.090 (9.58)	0.091 (9.82)
femur	0.073 (10.55)	0.091 (22.21)
tibia	0.060 (9.82)	0.102 (35.00)
humerus	0.102 (14.55)	0.068 (9.41)
ulna	0.395 (24.91)	0.075 (11.59)
radius	0.395 (23.60)	0.075 (10.98)

4.1.2. B: 2D Pose Based Kinematic Estimation

The results of the comparison of our proposed method, CMS, OpenPose, and MediaPipe are shown in Table 3. We find that algorithms trained on noisy labels, that use fewer key points perform worse than ours. We see a clear difference between the unsmoothed OpenPose estimations and the smoothed MediaPipe estimations in the mean velocity of the estimations. Our proposed method D3KE still performs better showing that even in an ideal scenario (CMS, i.e., no noise in the labels, enough markers, same distribution training data) direct estimation is preferable.

Table 3. Comparison of popular pose estimation algorithms to D3KE. As OpenPose and MediaPipe require multiple cameras to create 3D keypoints, we compare against the average of both camera views for CMS and D3KE. CMS shows better performance than OpenPose and MediaPipe and D3KE shows the overall best performance. Indicating that direct estimation is preferable to (naive) implementations of multi-step methods.

	MAE_{angle} (°)	SD_{angle} (°)	MV_{angle} (°/s)
OpenPose	16.98	25.91	75.15
MediaPipe	10.60	18.80	37.15
CMS	5.11	10.27	15.74
D3KE	3.41	6.05	13.57

4.2. Sequential Network Variants

Table 1 also shows the results of different sequential networks for smoothing of the predictions. Although the convolutional model by itself already has good performance in joint angle and scale factor estimation, using temporal smoothing can additionally reduce the estimation error.

The results of our investigation to reduce the noise in the estimations using temporal smoothing are shown in Table 4. The result shows that all temporal models can improve the smoothness of the sequence. The LSTM achieves the best performance on MV_{BL} and MV_{angle} among all temporal models, this is contrary to the results in Table 1, in which using a Transformer as the sequential model yielded the best results.

Table 4. The mean velocity errors for bony landmarks MV_{BL} and joint angles MV_{angle}, lower values indicate smoother estimations. Adding a sequential model most probably improves the continuity of estimations.

	MV_{BL} (mm/s)	MV_{angle} (°/s)
Convolutional model	378.01	21.7
Conv. + LSTM	243.23	12.19
Conv. + TCNs	245.35	12.29
Conv. + Transformer	262.82	13.57

4.3. Processing Speed

Our proposed method achieves 31.96 fps with a batch size of 256, as shown in Table 5. Since the skeletal-model layer must traverse body segments in the level order, our proposed method is slower than the CMS for a batch size of 1. However, the support of mini-batch computation in the skeletal-model layer allows D3KE to run faster than the CMS. Showing that our method can reach video framerate speeds on a competent GPU.

Table 5. Comparison of processing speed in FPS of D3KE and the baseline for multiple images or 'batches' in parallel. OpenSim does show little change in processing speed for increasing batch sizes. The proposed method achieves framerate speeds for batches of 256 images, allowing it to analyze images as fast as a common webcam or mobile phone camera collects them. Bold numbers indicate best performance.

Batch Size	1	16	64	128	256
D3KE	0.92	8.78	20.94	28.25	**31.96**
CMS	7.51	8.36	8.43	**8.44**	8.35

4.4. Generalization Performance

Figure 4 shows that both CMS and D3KE have relatively little variation across different movements and different participants, yet larger variations across individual joints. This is also reflected in median MAEs and ρ per joint (Table 6), with median MAEs for joints varying within a range 3.8° for joints, while movements and participants vary under 1°. It shows that D3KE generalizes well to different participants and movements.

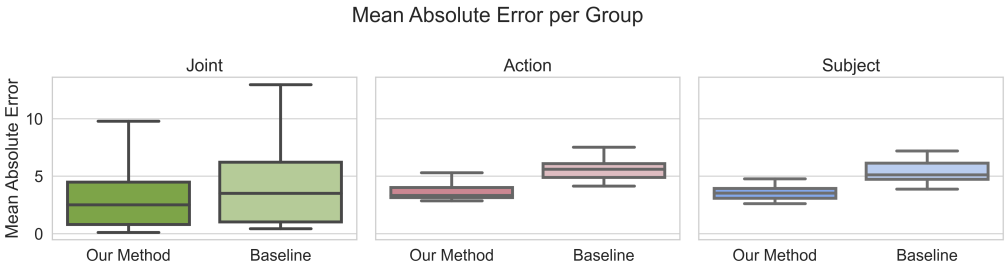

Figure 4. Mean absolute error for predicted joint angles per joint, movement and subset. Across these groups, D3KE shows less variation compared to the CMS. The low variations indicate that D3KE is suitable for use on different participants and movements.

Table 6. Median and inter-quartile ranges (IQR) of joint angles per *joint*, *movement* and *participant*. MAE, RMSE and correlation are calculated over individual frames, Medians and IQR are reported due to the skewed distribution of results. Within each group, both camera views show similar errors. Joint angles show the highest error and highest spread of values of all groupings. D3KE generalizes well to different movements, participants and camera views.

Group	Camera View	MAE (°)		RMSE (°)		ρ	
		Median	IQR	Median	IQR	Median	IQR
Joint	Frontal	2.13	3.80	2.54	3.96	0.77	0.16
	Sagittal	2.14	3.03	2.55	3.49	0.73	0.21
Movement	Frontal	1.85	0.63	2.19	0.84	0.76	0.11
	Sagittal	1.91	0.46	2.30	0.65	0.74	0.11
Participant	Frontal	1.76	0.53	2.03	0.66	0.77	0.04
	Sagittal	1.84	0.28	2.14	0.35	0.74	0.04

4.5. Qualitative Results

We visualize the estimation of musculoskeletal models from our proposed method with the Transformer architecture in Figure 5. We also show the comparison between estimation and ground truth of the left knee angle as an example of joint angle estimation quality. We took the average body scales of the predicted sequence of scaling factors to scale the model and visualize it in OpenSim using the predicted joint angles as inputs. From the figure, we can see that the proposed method can achieve results that are in agreement with the single-view input video.

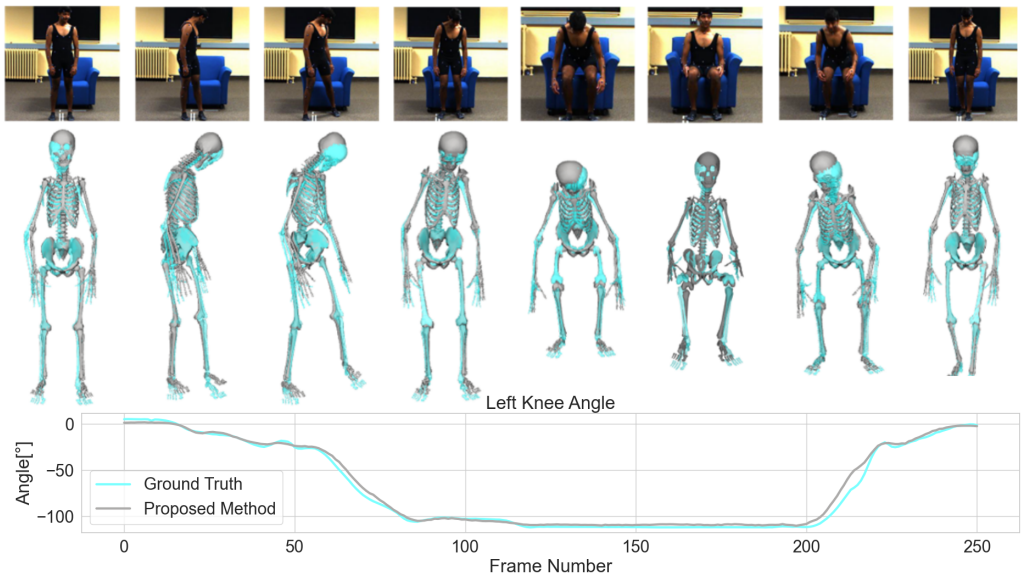

Figure 5. Qualitative results of D3KE from a 'sitting-down' movement in the BML-Movi dataset. The top row shows selected frames throughout the movement. The middle row shows different poses of the ground truth skeletal model throughout the movement (cyan) and the skeletal model (white) based on D3KE's estimation. The bottom row shows the changes in flexion/extension of the left knee throughout the movement with blue being the predicted and orange being the ground truth angle.

5. Discussion

In summary, we compared a direct approach of estimating joint angles from video images to the more traditional multi-step approach found in most recent works. The traditional method first estimates key points from a video of a subject, then calculates joint angles using a (musculo)skeletal model through an inverse-kinematics process. We developed a method consisting of a convolutional neural network and a sequential network both including a specialized layer that performs kinematic transforms of a (musculo)skeletal model and allows for direct optimization of the predicted joint angles (D3KE) and treats the prediction of key points only as an auxiliary task. We compared our direct estimation approach against naive implementations of often used algorithms in the related literature, as well as a self-implemented custom multi-step approach (CMS) that is trained on the same data as our direct approach. We show that direct estimation of kinematics yields higher accuracy in predicted joint angles compared to the traditional multi-step approach. Our results indicate that direct estimation can help the future development of algorithms for fast and accessible kinematic analysis for researchers and clinicians.

5.1. Direct vs. Multi-Step Estimation

5.1.1. 3D-Pose Kinematic Estimation

To compare direct estimation vs. multi-step estimation, we compared our D3KE method against a 3D-pose based multi-step approach (CMS) with comparable network architecture and trained on the same training data. Compared to the CMS, our proposed method improves the accuracy of joint angle estimation. For all model combinations, we can see an improvement of about 35% in accuracy for the estimation of joint angles. Our results support the feasibility of our proposed method. It delivers improvements due to directly optimizing the predicted joint angles and scaling factors while using the pose estimates only as an auxiliary task. The auxiliary task effectively imposes a constraint on the network estimation. We show that direct optimization is preferable to the multi-step approach when using videos from a single-camera view. We expect that using additional specialized layers, a network might be able to directly optimize for individual muscle forces with comparable accuracy from a monocular video.

5.1.2. 2D-Pose Kinematic Estimation

We compared our direct approach (D3KE) and the self-trained multi-step approach (CMS) against two multi-step approaches commonly used in the literature. Compared to the more traditional implementations of the OpenPose and MediaPipe algorithms, both our proposed and our CMS method show superior performance. From Tables 4 and 6, we see that estimations from D3KE are far smoother compared to the traditional methods. This is likely due to multiple reasons. The predicted key points of OpenPose suffer from systematic errors due to inaccuracies in their training data [9], which can explain the drop in performance, for MediaPipe the accuracy of labels in their training data is not known as their paper only states that their annotators were human [59], not whether they had expertise in labeling anatomical key points. The lack of smoothing for the predicted OpenPose key points can also contribute to its overall worse performance. MediaPipe, which uses internal smoothing, shows better results in comparison. In general, we expected worse performance from OpenPose and MediaPipe as they only predict 18 or 33 key points respectively, while we supervise a total of 77. Both traditional methods predict keypoints representing joint centers and not markers on body segments, this makes it hard to distinguish rotations between different body segments during the musculoskeletal modeling step.

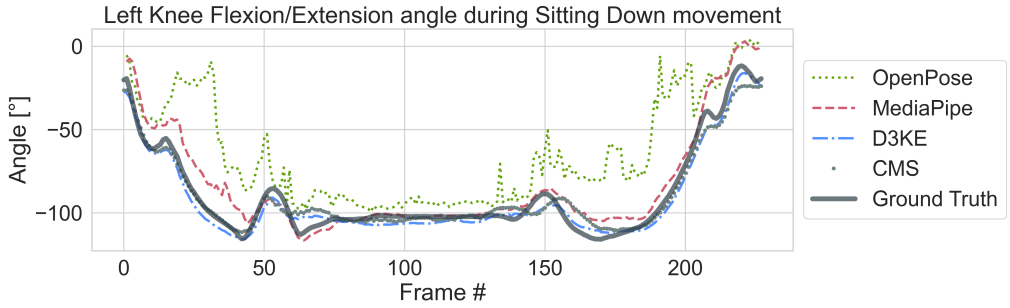

Figure 6. Qualitative comparison of predicted angles of the left knee, from a 'sitting-down' movement in the BML-Movi dataset. While OpenPose shows by far the noisiest estimation, the smoothing of the MediaPipe estimation is clear, our proposed method and implemented CMS work best, probably due to the restriction of additional markers.

5.2. Network Variants

We compared three commonly used sequential networks to improve our estimation. We show that all sequential networks improve our estimation accuracy. One possible explanation for this is that the network is better able to handle 'self-occlusion' artifacts. The estimation of a 3D-pose from a single camera is an ill-posed problem as a 3D point projected to a 2D image can originate from any position along the ray(s) that fall on the image sensor and form the corresponding pixel. This can lead to self-occlusion, such as the torso and left arm occluding the right arm during a right arm swing in frames recorded from the left sagittal view. During self-occlusion, it is difficult for frame-based networks to make a good estimate as they lack temporal information of previous angles of the arm to extrapolate from. Sequential networks on the other hand have access to temporal information, which can allow for more accurate estimations. Although we only see a slight increase 0.001° in MAE of the joint angle estimation, we can see a clear improvement of the sequential models in the smoothness of the predicted angles Table 4. This might be due to the network learning to interpolate motions during occurrences of self-occlusion.

5.3. Processing Speed

Compared to the multi-step baseline, CMS, our D3KE approach shows increased calculation speeds for larger batch sizes. Both CMS and D3KE make use of the same ResNeXt50 architecture, which should show approximately the same performance increase

with increasing batch sizes for both methods. D3KE could be expected to be slower, as it also has the additional time cost of calculating the pose from the estimated kinematics in the skeletal model layer. However, due to its multi-step nature, CMS has to perform an additional inverse kinematics calculation. This calculation seems to form a bottleneck in the processing speed of the CMS approach restricting it to a framerate of 8 fps. Other multi-step algorithms will most likely encounter the same problem. In the case of OpenPose, which runs at about 4 fps [59], even lower frame rates can be expected for a complete pipeline. This shows the advantage in the processing time of our direct approach.

For a method to be usable in everyday life, it should be reasonably fast in running. Processing speeds allowing a method to run between 15 and 30 frames per second are favorable, as they show that a method can process a video as fast as its frames are collected. However, our results might not directly translate to every real-world scenario. To process multiple images simultaneously as batches, D3KE currently requires GPUs that are not available in mobile devices, which prevents it from beingportable. In addition, we use the Faster R-CNN object detection network to crop our images. This step was not included in the processing speed evaluation, as it is highly dependent on the chosen object-detection algorithm. However, with inference speeds of 12fps, the Faster R-CNN object detection would form a bottleneck in applying our method in real-time applications. Given the speed of development in the field of object detection, Faster R-CNN can by now be regarded as an old algorithm, and newer and faster object detectors should be used instead. The YoloV7 algorithm [63], which performs object detection at up to 286 fps could be considered. In general, the current architecture is not optimized for speed or a specific technology and we are using off-the-shelf, fairly standard convolutional and sequential architectures. For these architectures, smaller and faster alternatives might be found in the future. When optimizing for all these points, we predict the proposed method could run on mobile devices within a few years, effectively enabling a 3D kinematic analysis instrument to become available for everyone with a mobile phone or tablet.

5.4. Generalization Performance

Our method generalizes well on the tested data. As shown in Figure 4 and Table 6, the estimation variations across participants and movements are small. We can conclude that our method shows the ability to generalize to different camera views, participants, and performed movements, within the tested dataset. Our results indicate that D3KE could be generally applied to a variety of people and movements, including clinical and sports applications, e.g., physiotherapists and athletes, when trained on sufficient additional data.

Although we show good generalization performance on the BMLMovi database, it is difficult to estimate how well our method will generalize in a real-world scenario. In machine learning settings, training data is often not representative of the task of the network in the real world [64] and can introduce biases if applied to scenarios that are very different from the one represented in the training data. Unfortunately, there is currently a lack of deep learning datasets for kinematic analysis [8–10,20]. In addition, while the BML-Movi database is excellent for training neural networks due to the large number of participants performing movements and the diversity of execution styles, it might be not extensive enough to train a network for biomedical applications in the real world. However, to evaluate the current method fully, such an extensive dataset would be necessary. In general, we expect a drop in accuracy when our method is applied to a scenario different from the BMLMovi database. As we train on just two calibrated cameras, we expect our method to be most vulnerable to alternative camera positions, that do not show people in either frontal or sagittal view. Future research should investigate the stability of direct estimation methods when applied to data that differs significantly from the training data.

5.5. Future Work

To improve the accuracy of the algorithm and provide further insight into the strengths and weaknesses of monocular joint angle estimation, a new dataset with dedicated annota-

tion is needed. A dataset specifically designed for the estimation of joint angles and/or kinetics could improve the accuracy of the algorithm. This dataset could be established with a large number of camera views, and top-down views for better estimation of movements in the transverse plane, where participants perform movements that exercise the full ROMs of individual joints including upper extremities, as well as movements that are relevant for health care professionals such as physiotherapy exercises and other clinical tests. In addition, the inclusion of abnormal movement patterns could give better insights into the clinical relevance of newly developed methods.

Transfer learning could be explored to apply 3DKE in settings where little training data are available. Vdeos that are very different from the BMLMovi training data, such as people wearing more clothes, are in different surroundings, or are filmed from a different camera view, will, most probably, yield worse accuracy than shown in this paper. Transfer learning of a pre-trained D3KE on a minimal portion of a dataset could be investigated as an alternative to the time-consuming collection of a novel dataset.

The capabilities of D3KE as an adapter for kinetic analysis of a movement in OpenSim could be explored. Given data similar to BMLMovi or successful transfer learning on relevant data beforehand, our method provides an easy way to skip the tedious steps of scaling and running inverse kinematics on an MSM. This enables the quick generation of MSMs for kinetic analysis from just a single video. Even if this kinematic estimation comes at the cost of reduced accuracy, it could provide coarse insights into collected data, which can later be confirmed through finer analysis with the manually scaled MSMs.

D3KE could be made more generally applicable if the underlying model of the Skeletal-model layer would not be fixed. Currently, the underlying model is fixed in the Skeletal-model layer. Future iterations could explore combinations of the Pytorch and OpenSim python libraries to allow training a network on a self-defined model or allowing a pre-trained model to be refined through transfer learning for, e.g., only estimation of joint angles around the shoulder.

Existing Explainable AI tools should be applied to better understand the inner workings of D3KE. Deep neural networks are capable of high accuracy estimation, because of their ability to break down highly complex tasks into simpler tasks [24], but understanding what these simpler tasks are is non-trivial. Research in Explainable AI has generated tools and frameworks that allow one to better understand the basis of the final predictions of a network [65]. Applying these tools could help users and researchers alike to better understand the biases and limitations of our method. D3KE can still predict the joint angles even if these joints are occluded; this means it must make assumptions. What these assumptions are and how they came to be are important to estimate the trustworthiness of this algorithm in a real-world scenario.

6. Conclusions

In this paper, we present a novel end-to-end neural network for the estimation of segment joint angles of the human body. Compared to the previous method, we directly regress to the joint angle and scale for individual segments from the input video. We trained our method from scratch on the BML-Movie database and compared it against a 3D pose estimation method on which we used the inverse kinematics tool of OpenSim to obtain the kinematics.

We conclude that using direct estimation of joint angles is preferable in a single camera setting, as it is more accurate compared to the common approach of fitting an estimated pose to a musculoskeletal model and performing inverse kinematics. By allowing the network to directly optimize for the joint angles and scaling factors, our method is less prone to errors in the key point labels used to predict key point location for pose estimation. In addition, the use of a sequential model is important when designing a neural network architecture for kinematic estimation, as it allows to smooth predictions over time to create better estimates of limb position and joint angles during self-occlusion.

While using deep learning for biomedical solutions is still in its infancy, the presented method shows that training networks from scratch for specialized tasks is a viable way to estimate joint angles from a single camera video. With further advancements in the underlying algorithms as well computational performance, we predict that the methodology we have presented will assist biomedical and clinic practitioners to measure and monitor human movement in the near future.

Author Contributions: Conceptualization, M.B. and W.-T.Y.; methodology, M.B. and W.-T.Y.; software, W.-T.Y.; validation, M.B. and W.-T.Y.; formal analysis, M.B. and W.-T.Y.; investigation, M.B. and W.-T.Y.; resources, X.Z., J.v.G. and F.C.T.v.d.H.; data curation, W.-T.Y.; writing—original draft preparation, W.-T.Y., M.B. and X.Z.; writing—review and editing, M.B., A.S., X.Z., J.v.G. and F.C.T.v.d.H.; visualization, M.B. and W.-T.Y.; supervision, X.Z., A.S., J.v.G. and F.C.T.v.d.H.; project administration, M.B.; funding acquisition, M.B. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Dutch Research Council (NWO) under the Citius Altius Sanius Perspective Program P16-28 Project 4.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The publicly available BMLMovi database was obtained from the BioMotion Lab at York University and is available via https://www.biomotionlab.ca/movi/ (accessed on 17 May 2022).

Acknowledgments: The authors would like to thank Lisa Noteboom for her feedback as well as Marco Hoozemans and Dirkjan Veeger for guidance and insight during our bi-weekly meetings.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CMS	Custom multi-stage approach
D3KE	Direct 3D kinematic estimation
IQR	Interquartile Range
MAE	Mean absolute error
MMC	Markerless motion capture
MPBLPE	Mean per bony landmark error
MSM	Musculoskeletal model
MVE	Mean velocity error
OMC	Optical Motion Capture
PCC	Pearson correlation coefficient
RMS	Root mean square error
ROM	Range of motion
SD	Standard Deviation
TCN	Temporal Convolutional Network

References

1. Kidziński, Ł.; Yang, B.; Hicks, J.L.; Rajagopal, A.; Delp, S.L.; Schwartz, M.H. Deep neural networks enable quantitative movement analysis using single-camera videos. *Nat. Commun.* **2020**, *11*, 4054. https://doi.org/10.1038/s41467-020-17807-z.
2. Pagnon, D.; Domalain, M.; Reveret, L. Pose2Sim: An End-to-End Workflow for 3D Markerless Sports Kinematics—Part 1: Robustness. *Sensors* **2021**, *21*, 6530. https://doi.org/10.3390/s21196530.
3. Pagnon, D.; Domalain, M.; Reveret, L. Pose2Sim: An End-to-End Workflow for 3D Markerless Sports Kinematics—Part 2: Accuracy. *Sensors* **2022**, *22*, 2712. https://doi.org/10.3390/s22072712.
4. Kanko, R.M.; Laende, E.K.; Strutzenberger, G.; Brown, M.; Selbie, W.S.; DePaul, V.; Scott, S.H.; Deluzio, K.J. Assessment of spatiotemporal gait parameters using a deep learning algorithm-based markerless motion capture system. *J. Biomech.* **2021**, *122*, 110414. https://doi.org/10.1016/j.jbiomech.2021.110414.

5. Gu, X.; Deligianni, F.; Lo, B.; Chen, W.; Yang, G. Markerless gait analysis based on a single RGB camera. In Proceedings of the 2018 IEEE 15th International Conference on Wearable and Implantable Body Sensor Networks (BSN), Las Vegas, NV, USA, 4–7 March 2018; pp. 42–45, ISSN: 2376-8894. https://doi.org/10.1109/BSN.2018.8329654.
6. Liao, R.; Yu, S.; An, W.; Huang, Y. A model-based gait recognition method with body pose and human prior knowledge. *Pattern Recognit.* **2020**, *98*, 107069. https://doi.org/10.1016/j.patcog.2019.107069.
7. Noteboom, L.; Hoozemans, M.J.M.; Veeger, H.E.J.; Van Der Helm, F.C.T. Feasibility and validity of a single camera CNN driven musculoskeletal model for muscle force estimation during upper extremity strength exercises: Proof-of-concept. *Front. Sport. Act. Living* **2022**, *4*, 994221. https://doi.org/10.3389/fspor.2022.994221.
8. Seethapathi, N.; Wang, S.; Saluja, R.; Blohm, G.; Kording, K.P. Movement science needs different pose tracking algorithms. *arXiv* **2019**, arXiv:1907.10226.
9. Cronin, N.J. Using deep neural networks for kinematic analysis: Challenges and opportunities. *J. Biomech.* **2021**, *123*, 110460. https://doi.org/10.1016/j.jbiomech.2021.110460.
10. Wade, L.; Needham, L.; McGuigan, P.; Bilzon, J. Applications and limitations of current markerless motion capture methods for clinical gait biomechanics. *PeerJ* **2022**, *10*, e12995. https://doi.org/10.7717/peerj.12995.
11. Needham, L.; Evans, M.; Cosker, D.P.; Wade, L.; McGuigan, P.M.; Bilzon, J.L.; Colyer, S.L. The accuracy of several pose estimation methods for 3D joint centre localisation. *Sci. Rep.* **2021**, *11*, 20673. https://doi.org/10.1038/s41598-021-00212-x.
12. Cao, Z.; Hidalgo, G.; Simon, T.; Wei, S.E.; Sheikh, Y. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *arXiv* **2019**, arXiv:1812.08008.
13. Mathis, A.; Mamidanna, P.; Cury, K.M.; Abe, T.; Murthy, V.N.; Mathis, M.W.; Bethge, M. DeepLabCut: Markerless pose estimation of user-defined body parts with deep learning. *Nat. Neurosci.* **2018**, *21*, 1281–1289. https://doi.org/10.1038/s41593-018-0209-y.
14. Fang, H.S.; Xie, S.; Tai, Y.W.; Lu, C. RMPE: Regional Multi-Person Pose Estimation. **arXiv 2017**, arXiv:1612.00137. pp. 2334–2343.
15. Pavllo, D.; Feichtenhofer, C.; Grangier, D.; Auli, M. 3D human pose estimation in video with temporal convolutions and semi-supervised training. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019.
16. Seth, A.; Hicks, J.L.; Uchida, T.K.; Habib, A.; Dembia, C.L.; Dunne, J.J.; Ong, C.F.; DeMers, M.S.; Rajagopal, A.; Millard, M.; et al. OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement. *PLoS Comput. Biol.* **2018**, *14*, e1006223. https://doi.org/10.1371/journal.pcbi.1006223.
17. Uchida, T.K.; Seth, A. Conclusion or Illusion: Quantifying Uncertainty in Inverse Analyses From Marker-Based Motion Capture due to Errors in Marker Registration and Model Scaling. *Front. Bioeng. Biotechnol.* **2022**, *10*, 874725.
18. Della Croce, U.; Cappozzo, A.; Kerrigan, D.C. Pelvis and lower limb anatomical landmark calibration precision and its propagation to bone geometry and joint angles. *Med. Biol. Eng. Comput.* **1999**, *37*, 155–161. https://doi.org/10.1007/BF02513282.
19. Fonseca, M.; Armand, S.; Dumas, R.; Leboeuf, F.; Bergere, M.; Cândido, J. The Conventional Gait Model's Sensitivity to Lower-limb Marker Placement. *Sci. Rep.* **2022**, *12*, 14207.
20. Needham, L.; Evans, M.; Cosker, D.P.; Colyer, S.L. Can Markerless Pose Estimation Algorithms Estimate 3D Mass Centre Positions and Velocities during Linear Sprinting Activities? *Sensors* **2021**, *21*, 2889. https://doi.org/10.3390/s21082889.
21. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2015**, arXiv:1409.1556. https://doi.org/10.48550/arXiv.1409.1556.
22. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90.
23. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. https://doi.org/10.1038/nature14539.
24. Allen-Zhu, Z.; Li, Y. Backward Feature Correction: How Deep Learning Performs Deep Learning. *arXiv* **2021**, arXiv:2001.04413. https://doi.org/10.48550/arXiv.2001.04413.
25. Sarandi, I.; Linder, T.; Arras, K.O.; Leibe, B. Metric-Scale Truncation-Robust Heatmaps for 3D Human Pose Estimation. In Proceedings of the 2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020), Buenos Aires, Argentina, 6–20 November 2020. https://doi.org/10.1109/fg47880.2020.00108.
26. Mehta, D.; Sridhar, S.; Sotnychenko, O.; Rhodin, H.; Shafiei, M.; Seidel, H.P.; Xu, W.; Casas, D.; Theobalt, C. VNect. *ACM Trans. Graph.* **2017**, *36*, 1–14. https://doi.org/10.1145/3072959.3073596.
27. Nibali, A.; He, Z.; Morgan, S.; Prendergast, L. 3D Human Pose Estimation with 2D Marginal Heatmaps. *arXiv* **2018**, arXiv:1806.01484.
28. Pavlakos, G.; Zhou, X.; Derpanis, K.G.; Daniilidis, K. Coarse-to-Fine Volumetric Prediction for Single-Image 3D Human Pose. *arXiv* **2017**, arXiv:1611.07828.
29. Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; He, K. Aggregated Residual Transformations for Deep Neural Networks. *arXiv* **2017**, arXiv:1611.05431.
30. Cheng, Y.; Yang, B.; Wang, B.; Tan, R.T. 3D Human Pose Estimation using Spatio-Temporal Networks with Explicit Occlusion Training. *arXiv* **2020**, arXiv:2004.11822.
31. Cheng, Y.; Yang, B.; Wang, B.; Yan, W.; Tan, R.T. Occlusion-Aware Networks for 3D Human Pose Estimation in Video *In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, South Korea, 27 October–2 November **2019**.
32. Liu, R.; Shen, J.; Wang, H.; Chen, C.; Cheung, S.-c.; Asari, V.K. Enhanced 3D Human Pose Estimation from Videos by using Attention-Based Neural Network with Dilated Convolutions. *arXiv* **2021**, arXiv:2103.03170.

33. Li, W.; Liu, H.; Ding, R.; Liu, M.; Wang, P. Lifting Transformer for 3D Human Pose Estimation in Video. *arXiv* **2021**, arXiv:cs.CV/2103.14304.
34. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.
35. Zhou, Y.; Barnes, C.; Lu, J.; Yang, J.; Li, H. On the Continuity of Rotation Representations in Neural Networks. *arXiv* **2020**, arXiv:1812.07035.
36. Lu, T.W.; O'Connor, J.J. Bone position estimation from skin marker co-ordinates using global optimisation with joint constraints. *J. Biomech.* **1999**, *32*, 129–134. https://doi.org/10.1016/S0021-9290(98)00158-4.
37. Al Borno, M.; O'Day, J.; Ibarra, V.; Dunne, J.; Seth, A.; Habib, A.; Ong, C.; Hicks, J.; Uhlrich, S.; Delp, S. OpenSense: An open-source toolbox for inertial-measurement-unit-based measurement of lower extremity kinematics over long durations. *J. Neuroeng. Rehabil.* **2022**, *19*, 22. https://doi.org/10.1186/s12984-022-01001-x.
38. Delp, S.L.; Anderson, F.C.; Arnold, A.S.; Loan, P.; Habib, A.; John, C.T.; Guendelman, E.; Thelen, D.G. OpenSim: Open-Source Software to Create and Analyze Dynamic Simulations of Movement. *IEEE Trans. Biomed. Eng.* **2007**, *54*, 1940–1950. https://doi.org/10.1109/TBME.2007.901024.
39. ANDERSON, F.C.; PANDY, M.G. A Dynamic Optimization Solution for Vertical Jumping in Three Dimensions. *Comput. Methods Biomech. Biomed. Eng.* **1999**, *2*, 201–231. https://doi.org/10.1080/10255849908907988.
40. Anderson, F.C.; Pandy, M.G. Dynamic Optimization of Human Walking . *J. Biomech. Eng.* **2001**, *123*, 381–390. https://doi.org/10.1115/1.1392310.
41. Delp, S.; Loan, J.; Hoy, M.; Zajac, F.; Topp, E.; Rosen, J. An interactive graphics-based model of the lower extremity to study orthopaedic surgical procedures. *IEEE Trans. Biomed. Eng.* **1990**, *37*, 757–767. https://doi.org/10.1109/10.102791.
42. Holzbaur, K.R.S.; Murray, W.M.; Delp, S.L. A model of the upper extremity for simulating musculoskeletal surgery and analyzing neuromuscular control. *Ann. Biomed. Eng.* **2005**, *33*, 829—840. https://doi.org/10.1007/s10439-005-3320-7.
43. Yamaguchi, G.T.; Zajac, F.E. A planar model of the knee joint to characterize the knee extensor mechanism. *J. Biomech.* **1989**, *22*, 1–10. https://doi.org/10.1016/0021-9290(89)90179-6.
44. Bruno, A.G.; Bouxsein, M.L.; Anderson, D.E. Development and Validation of a Musculoskeletal Model of the Fully Articulated Thoracolumbar Spine and Rib Cage. *J. Biomech. Eng.* **2015**, *137*, 081003. https://doi.org/10.1115/1.4030408.
45. Bruno, A.G.; Burkhart, K.; Allaire, B.; Anderson, D.E.; Bouxsein, M.L. Spinal Loading Patterns From Biomechanical Modeling Explain the High Incidence of Vertebral Fractures in the Thoracolumbar Region. *J. Bone Miner. Res.* **2017**, *32*, 1282–1290. https://doi.org/10.1002/jbmr.3113.
46. Burkhart, K.; Grindle, D.; Bouxsein, M.L.; Anderson, D.E. Between-session reliability of subject-specific musculoskeletal models of the spine derived from optoelectronic motion capture data. *J. Biomech.* **2020**, *112*, 110044. https://doi.org/10.1016/j.jbiomech.2020.110044.
47. Gonzalez, R.; Buchanan, T.; Delp, S. How muscle architecture and moment arms affect wrist flexion-extension moments. *J. Biomech.* **1997**, *30 7*, 705–712.
48. Jose Alejandro Amezquita Garcia. Modification of Wrist Model to include all the movements of the fingers. Available online: https://simtk.org/projects/moving-fingers (accessed on 16 June 2021).
49. Loper, M.; Mahmood, N.; Black, M.J. MoSh: Motion and Shape Capture from Sparse Markers. *ACM Trans. Graph.* **2014**, *33*. https://doi.org/10.1145/2661229.2661273.
50. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv* **2016**, arXiv: 1506.01497.
51. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. *arXiv* **2015**, arXiv:1512.03385.
52. Loshchilov, I.; Hutter, F. Decoupled Weight Decay Regularization. *arXiv* **2019**, arXiv:1711.05101.
53. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L. Pytorch: An imperative style, high-performance deep learning library. In Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2019; Volume 32.
54. Vryniotis, V.; Meier, P.; Hug, N.; Massa, F.; vfdev 5. torchvision. 2021. Available online: https://github.com/pytorch/vision (accessed on 20 September 2021).
55. Ghorbani, S.; Mahdaviani, K.; Thaler, A.; Kording, K.; Cook, D.J.; Blohm, G.; Troje, N.F. MoVi: A large multi-purpose human motion and video dataset. *PLoS ONE* **2021**, *16*, e0253157. https://doi.org/10.1371/journal.pone.0253157.
56. Mahmood, N.; Ghorbani, N.; Troje, N.F.; Pons-Moll, G.; Black, M.J. AMASS: Archive of Motion Capture as Surface Shapes. In Proceedings of the International Conference on Computer Vision, Seoul, South Korea, 27 October–2 November 2019; pp. 5442–5451.
57. Zhang, A.; Lipton, Z.C.; Li, M.; Smola, A.J. Dive into deep learning. *arXiv* **2021**, arXiv:2106.11342.
58. Hzzone. pytorch-openpose. Available online: https://github.com/Hzzone/pytorch-openpose (accessed on 8 September 2022).
59. Bazarevsky, V.; Grishchenko, I.; Raveendran, K.; Zhu, T.; Zhang, F.; Grundmann, M. BlazePose: On-device Real-time Body Pose tracking. *arXiv* **2020**, arXiv:2006.10204 , https://doi.org/10.48550/arXiv.2006.10204.
60. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2017**, arXiv:1412.6980.
61. Taylor, R. Interpretation of the correlation coefficient: A basic review. *J. Diagn. Med. Sonogr.* **1990**, *6*, 35–39.
62. Van Rossum, G.; Drake, F.L. *Python 3 Reference Manual*; CreateSpace: Scotts Valley, CA, USA, 2009.

63. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv* **2022**, arXiv.2207.02696.
64. Attenberg, J.; Ipeirotis, P.; Provost, F. Beat the Machine: Challenging Humans to Find a Predictive Model's "Unknown Unknowns". *J. Data Inf. Qual.* **2015**, *6*, 1:1–1:17. https://doi.org/10.1145/2700832.
65. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 618–626.