Supplementary Material for "Contact and Human Dynamics from Monocular Video"

Davis Rempe[1,2] Leonidas J. Guibas[1] Aaron Hertzmann[2] Bryan Russell[2]
Ruben Villegas[2] Jimei Yang[2]

[1]Stanford University
[2]Adobe Research

1 Introduction

This document includes additional details and results omitted from the main paper. Additionally, we urge the reader to **view the supplementary video** to fully appreciate the quality of 3D human motions produced by our method. We offer additional details of our methods in Sections 2, 3, 4, and 5, followed by additional evaluation details and experimental results in 6.

2 Contact Estimation Details

In this section we describe details of our contact estimation network and the new synthetic dataset used for training. Please refer to Section 3.2 of the main paper for the primary discussion.

2.1 Synthetic Dataset

Our synthetic dataset was rendered using Blender[1] and includes 13 characters performing 65 different motion capture sequences retargeted to each character taken from `www.mixamo.com`. Each motion is recorded from 2 camera viewpoints resulting in 1690 videos and 101k frames of data. The motions include: samba, swing, and salsa dancing, boxing, football, and baseball actions, walking, and idle poses. Videos are rendered at 1280x720 with motion blur, and are 2 seconds long at 30 fps. Example frames from the dataset are shown in Figure 1. In addition to RGB frames, at each timestep the dataset includes 2D OpenPose [1] detections, the 3D pose in the form of the character's skeleton (skeletons are different for each character, and pose is provided in a .bvh motion capture file), foot contact labels for the heel and toe base of each foot as described in the main paper, and camera parameters.

For each video, many parameters are randomized. The camera is placed at a uniform random distance in $[4.5, 7.5]$ and Gaussian random height with $\mu = 0.9$, and $\sigma = 0.3$ but clamped to be in $[0.3, 1.75]$, all in meters. The camera is placed at a random angle within 90 degrees offset from the front of the character but always looks at roughly character hip height. The camera does not move during the video. Floor texture is randomly chosen from solid colors and 26 other textures with various wood, grass, tile, metal, and carpet. Four lights in the scene are randomly toggled on and off, given random energies, and randomly offset from default positions, resulting in many shadow variations across videos.

[1]https://www.blender.org/

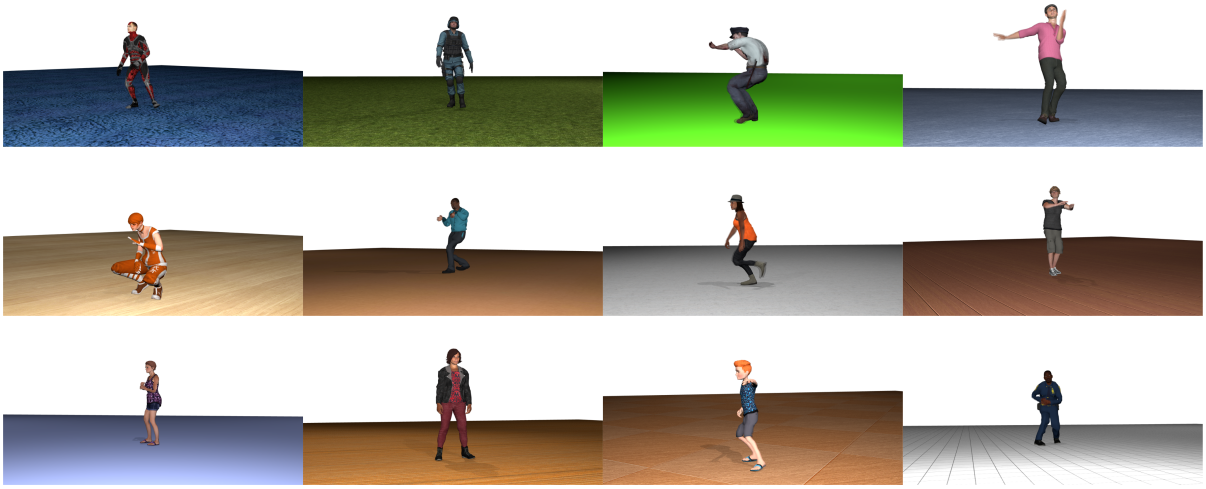

Figure 1: Example RGB frames from our synthetic dataset for contact estimation learning. The data also contains 2D OpenPose [1] detections, 3D pose in the form of the character's skeleton, and automatically labeled contacts for the toe base and heels.

2.2 Model Details

As discussed in the main paper, our model takes 2D pose as input which enables training on synthetic data. In future work, we would like to use image inputs to make contact and ground estimation more generally robust. For example, our model could be used to generate noisy pseudo-labels to bootstrap training on real images. One alternative to using 2D joints as input is to use 3D joints from Monocular Total Capture (MTC) [14], however the 3D MTC joints are derived directly from 2D OpenPose estimates and therefore give no new information to the network.

We implement our contact estimation MLP (sizes 1024, 512, 128, 32, 20) in PyTorch [8]. All but the last layer are followed by batch normalization, and we use a single dropout layer before the size-128 layer (dropout $p = 0.3$). To train, we optimize the binary cross-entropy loss using Adam [6] with a learning rate of 10^{-4}. We apply an L2 weight decay with a weight of 10^{-4} and use early stopping based on the validation set loss. We scale all 2D joint inputs to be largely between $[-1, 1]$. During training, we also add Gaussian noise to the normalized joints with $\sigma = 0.005$.

Because our network classifies contacts jointly over 5 frames for every target frame (the frame at the center of the window), there are many overlapping predictions at test time. When inferring contacts for an entire video at test time, we first use every frame as a target and then collect votes from overlapping predictions. A joint is marked in contact at a frame if a majority of the votes for that frame classify it as in contact.

3 Kinematic Optimization Details

In this section we detail the kinematic optimization procedure used to initialize our physics-based optimization. Please see Section 3.3 in the main paper for an overview.

3.1 Inputs and Initialization

Our kinematic optimization takes in information about J body joints over T timesteps that make up the motion. Specifically, for $j \in J$ and $t \in T$ we have the 2D pose detection from OpenPose $\mathbf{x}_{j,t} \in \mathbb{R}^2$ with

confidence $\sigma_{j,t}$, contacts estimated in the previous stage of our pipeline $c_{j,t} \in \{0, 1\}$ where $c_{j,t} = 1$ if joint j is in contact with the ground at time t, and finally the full-body 3D pose from MTC.

We use the MTC input to initialize a custom skeleton, called $\mathbf{S}_{src}$ in the main paper, which contains $J = 28$ joints. In particular, we use the MTC COCO regressor to obtain 19 body joint positions (excluding feet) over the sequence, and vertices on the Adam mesh for the 6 foot joints as in the original MTC paper. We choose these 25 joints in order to use a re-projection error in our optimization objective, as described below. To better map to the character rigs used during animation retargeting, we additionally use 3 spine joint positions directly from the MTC body model giving the final total of 28 joints. Note, we do not use any hand or face information from MTC. Our skeleton has fixed bone lengths which are determined based on these input 3D joint locations throughout the motion sequence: the median distance between two joints over the entire motion sequence defines the bone length. Our skeleton (before fitting bone lengths to the input data) is visualized in Figure 2.

We normalize the input positions to get the root-relative positions of each joint $\overline{\mathbf{q}}_{j,t} \in \mathbb{R}^3,\ j = 1, \ldots, J,\ t = 1, \ldots, T$ which we will target during optimization, and let the global translation be our initial root translation $\overline{\mathbf{p}}_{root,t}$. All these positions are preprocessed to remove obviously incorrect frames based on OpenPose detection confidence: for the 25 joints with corresponding 2D OpenPose detections (all non-spine joints in our skeleton), if the confidence is below 0.3, then the frame is replaced with a linear interpolation between the closest adjacent frames with sufficient confidence.

Because we optimize for the joint angles of our skeleton (see below), next we must find initial joint angles to match the MTC joint position inputs. We roughly initialize the joint angles of our skeleton by copying those from the MTC body model, and finally perform inverse kinematics (IK) targeting the preprocessed joint positions which results in a reconstruction of the MTC input on our skeleton. We use a Jacobian-based full body IK solver based on [3]. This is the skeleton which is optimized throughout our kinematic initialization.

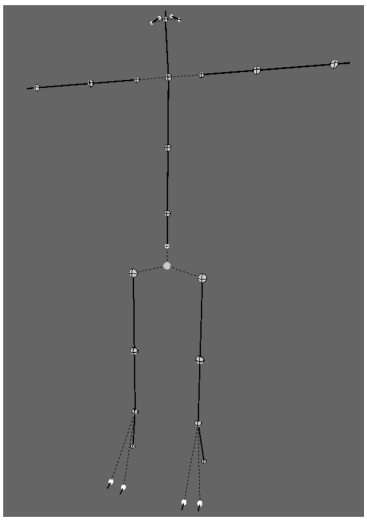

Figure 2: Skeleton used in our method. Bone lengths and pose are initialized from MTC input for each motion sequence before our kinematic optimization.

3.2 Optimization Variables

We optimize over global 3D root translation $\mathbf{p}_{root,t} \in \mathbb{R}^3$ and skeleton joint Euler angles $\boldsymbol{\theta}_{j,t} \in \mathbb{R}^3$ with $j = 1, \ldots, J,\ t = 1, \ldots, T$. We also find ground plane parameters $\hat{\mathbf{n}}, \mathbf{p}_{floor} \in \mathbb{R}^3$ which are the normal vector and some point on the plane. As described below, we do not jointly optimize all of these at once; we do it in stages and fit the floor separately.

3.3 Problem Formulation

We seek to minimize the following objective function:

$$\alpha_{proj}E_{proj} + \alpha_{vel}E_{vel} + \alpha_{ang}E_{ang} + \alpha_{acc}E_{acc} + \alpha_{data}E_{data} + \alpha_{cont}E_{cont} + \alpha_{floor}E_{floor} \tag{1}$$

where the α are constant weights. We use $\alpha_{proj} = 0.5$, $\alpha_{vel} = \alpha_{ang} = 0.1$, $\alpha_{acc} = 0.5$, $\alpha_{data} = 0.3$, and $\alpha_{cont} = \alpha_{floor} = 10$. We now detail each of these energy terms.

Suppose $\mathbf{q}_{j,t} \in \mathbb{R}^3,\ j = 1, \ldots, J,\ t = 1, \ldots, T$ are the current root-relative joint position estimates during optimization which can be calculated using forward kinematics on $\mathbf{S}_{src}$ with the current joint angles $\boldsymbol{\theta}_{j,t}$. Then our energy terms are defined as follows.

- *2D Re-projection Error*: minimizes deviation of joints from corresponding OpenPose detections, weighted by detection confidence

$$E_{proj} = \sum_{t=1}^{T}\sum_{j=1}^{J} \sigma_{j,t}||\Pi(\mathbf{q}_{j,t} + \mathbf{p}_{root,t}) - \mathbf{x}_{j,t}||^2 \tag{2}$$

where Π is the perspective projection parameterized by focal length f (assumed to be 2000) and $[c_x, c_y]$.

- *Velocity Smoothing*: minimizes change in joint positions and angles over time

$$E_{vel} = \sum_{t=1}^{T-1} \sum_{j=1}^{J} ||\mathbf{q}_{j,t+1} - \mathbf{q}_{j,t}||^2 \tag{3}$$

$$E_{ang} = \sum_{t=1}^{T-1} \sum_{j=1}^{J} ||\boldsymbol{\theta}_{j,t+1} - \boldsymbol{\theta}_{j,t}||^2. \tag{4}$$

- *Linear Acceleration Smoothing*: minimizes change in joint linear velocity over time

$$E_{acc} = \sum_{t=1}^{T-2} \sum_{j=1}^{J} ||(\mathbf{q}_{j,t+2} - \mathbf{q}_{j,t+1}) - (\mathbf{q}_{j,t+1} - \mathbf{q}_{j,t})||^2. \tag{5}$$

- *3D Data Error*: minimizes deviation from 3D MTC joint initialization

$$E_{data} = \sum_{t=1}^{T} \sum_{j=1}^{J} ||\mathbf{q}_{j,t} - \overline{\mathbf{q}}_{j,t}||^2. \tag{6}$$

- *Contact Velocity Error*: encourages feet joints (toes and heels) to be stationary when labeled as in contact

$$E_{cont} = \sum_{t=1}^{T-1} \sum_{j \in J_F} ||c_{j,t} \left((\mathbf{q}_{j,t+1} + \mathbf{p}_{root,t+1}) - (\mathbf{q}_{j,t} + \mathbf{p}_{root,t}) \right)||^2. \tag{7}$$

  where J_F is the set of foot joints.

- *Contact Position Error*: encourages toe and heel joints to be on the ground plane when labeled as in contact

$$E_{floor} = \sum_{t=1}^{T} \sum_{j \in J_F} ||c_{j,t} \left(\hat{\mathbf{n}} \cdot (\mathbf{q}_{j,t} + \mathbf{p}_{root,t} - \mathbf{p}_{floor}) \right)||^2. \tag{8}$$

3.4 Optimization Algorithm

We perform this optimization in three main stages. First, we enforce all objectives *except* the contact position error and solve only for skeleton root position and joint angles (no floor parameters). Next, we use a robust Huber regression to find the floor plane that best matches the foot joint contact positions and reject outliers, i.e., joints labeled as in contact when they are far from the ground. Outlier contacts are re-labeled as non-contacts for all subsequent processing. Finally, we repeat the full-body optimization, now enabling the contact position objective to ensure feet are on the ground plane. We optimize using the Trust Region Reflective algorithm with analytical derivatives.

3.5 Extracting Inputs for Physics-Based Optimization

From the full-body output of this kinematic optimization, we need to extract inputs for the physics-based optimization (Section 3.1 in the main paper). To get the COM targets $\overline{\mathbf{r}}(t) \in \mathbb{R}^3$, we treat each body part as a point with a pre-defined mass [4]. This also allows the calculation of the body-frame inertia tensor at each time step $\mathbf{I}_b(t) \in \mathbb{R}^{3\times3}$ which is used to enforce dynamics constraints. Unless otherwise noted, we assume a body mass of 73 kg for the character. We use the orientation about the root joint as the COM orientation $\overline{\boldsymbol{\theta}}(t) \in \mathbb{R}^3$ and the feet joint positions $\overline{\mathbf{p}}_{1:4}(t) \in \mathbb{R}^3$ are directly taken from the full-body motion.

4 Physics-Based Trajectory Optimization

In this section we detail various aspects of our reduced-dimensional physics-based trajectory optimization which is introduced in Section 3.1 of the main paper.

4.1 Polynomial Parameterization

COM position and orientation, foot positions during flight, and contact forces during stance are parameterized by a sequence of cubic polynomials as done in Winkler et al. [13]. These polynomials use a Hermite parameterization: we do not optimize over the polynomial coefficients directly, rather the duration, starting and ending positions, and boundary velocities.

The COM position and orientation use one polynomial every 0.1 s. Feet positions and forces always use at least 6 polynomials per phase, which we found necessary to accurately produce extremely dynamic motions. We adaptively add polynomials depending on the length of the phase. If the phase is longer than 2 s, additional polynomials are added commensurately. Foot positions during stance are a single constant value and contact forces during flight are constant 0 value. This ensures that the no slip and no force during flight constraints are met.

Please see Winkler et al. [13] for a more in-depth discussion of the polynomial parameterization along with the contact phase duration parameterization.

4.2 Constraint Parameters

Though the optimization variables are continuous polynomials, objective energies and constraints are enforced at discrete intervals. Leg and foot kinematic constraints are enforced at 0.08 s intervals, the above floor constraint at 0.1 s intervals, and dynamics constraints are enforced every 0.1 s. In practice, the velocity boundary constraints try to match the *mean* initial velocity over the first(last) 5 frames to make it more robust to noisy motion.

Objective terms, including smoothing, are enforced at every step for which we have input data. For example, the synthetic dataset at 30 fps will provide an objective term at $(1/30)$ s intervals.

4.3 Contact Timing Optimization

As explained in Section 3.1 of the main paper, our physics optimization is done in stages such that contact phase durations are not optimized until the very last stage. We found that allowing these durations to be optimized along with dynamics does not always result in a better solution as it is an inherently harder and less stable optimization. Therefore, in the presented results we use the better of the two solutions: either the solution using fixed input contact timings (from our neural network) or the solution after subsequently allowing phase durations to change, if the motion is improved.

4.4 Full-Body Output

Following the physics-based optimization, we must compute a full-body motion from the physically-valid COM and foot joint positions using IK. For the upper body (including the root), we calculate the offset of each joint from the COM in the input motion to the physics optimization, and use this offset added to the new optimal COM as the joint targets during IK. This means the upper-body motion will be essentially identical to the result of the kinematic optimization (though the posture may improve due to the new COM position). For the lower body, we target the toe and heel joints directly to the physically optimized output and let the remainder of the joints (i.e., ankles, knees, and hips) result from IK, which can be drastically different from the input. We use the same IK algorithm as in Section 3.1.

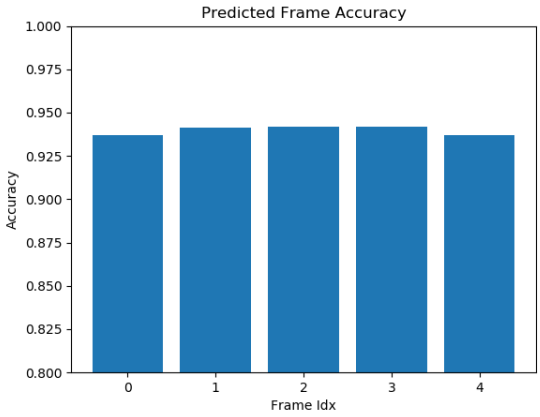

Figure 3: Contact estimation classification accuracy for all frames in the 5-frame output window on the synthetic test set (given 9 frames as input). The center frame index 2 is the target frame, however off-target contact predictions are still extremely accurate.

5 Retargeting to a New Character

In many cases, we wish to retarget the estimated motion to a new animated character mesh. We do this in the main paper in Section 4.2 for qualitative evaluation. One could apply physics-based motion retargeting methods to the output motion after an IK retargeting procedure, e.g., [12]. However, we avoid this extra step by directly performing our physics-based optimization on the target character skeleton.

Given a target skeleton $\mathbf{S}_{tgt}$, we insert an additional retargeting step following the kinematic optimization (see Figure 2 in the main paper). Namely, we uniformly scale $\mathbf{S}_{src}$ to the approximate size of our target skeleton, and then perform an IK optimization based on a predefined joint mapping to recover joint angles for $\mathbf{S}_{tgt}$. Then, the subsequent physics-based optimization and full-body upgrade are performed with this skeleton replacing $\mathbf{S}_{src}$. We use the same IK algorithm as in Section 3.1.

Note for qualitative comparison to MTC, we perform a very similar procedure: we first fit our skeleton to the raw MTC input, similar to as described in Section 3.1 but without the preprocessing, and then perform the IK retargeting described in this section. This provides a stronger baseline than a naive approach like directly copying joint angles from MTC to $\mathbf{S}_{tgt}$.

6 Evaluation Details and Additional Results

Here we include additional results and details which could not fit in the main paper.

6.1 Contact Estimation

Main results for contact estimation are presented in Section 4.1 of the main paper. One particularly interesting result is that using upper-body joints down to the knees as input to the network yields surprisingly high accuracy even on real data (0.865). This is a promising result for future work that may lead to accurate motion prediction even with occluded feet. Table 2 supplements the main results in the paper

Input **Window**	Prediction **Window**	Synthetic **Accuracy**	Real **Accuracy**
3	3	0.931	0.919
5	3	0.933	0.913
5	5	0.943	0.906
7	3	0.936	0.922
7	5	0.941	0.923
7	7	0.943	0.926
9	3	0.936	0.905
9	5	0.941	**0.935**
9	7	0.942	0.921
9	9	**0.946**	0.927

Table 1: Ablation study of input and output window sizes for learned contact estimation. Classification accuracy for many different combinations are shown.

Baseline Method	Synthetic Prec / Rec / F1	Real Prec / Rec / F1	MLP Input Joints	Synthetic Prec / Rec / F1	Real Prec / Rec / F1
Random	0.679 / 0.516 / 0.586	0.627 / 0.487 / 0.548	Upper to hips	0.940 / 0.941 / 0.940	0.728 / 0.837 / 0.779
Always Contact	0.677 / 1.000 / 0.808	0.647 / 1.000 / 0.786	Upper to knees	**0.958** / 0.946 / 0.952	0.926 / 0.859 / 0.892
2D Velocity	0.861 / 0.933 / 0.896	0.922 / 0.868 / 0.894	Lower to ankles	0.933 / 0.971 / 0.952	**0.963** / 0.916 / 0.939
3D Velocity	0.858 / 0.876 / 0.867	0.920 / 0.884 / 0.902	Lower to hips	0.941 / **0.973** / **0.957**	0.956 / **0.943** / **0.949**

Table 2: Precision, recall, and F1 Score (*Prec/Rec/F1*) of estimating foot contacts from video. Left: comparison to various baselines, Right: ablations using subsets of joints as features. Supplements Table 1 in the main paper.

	Dynamics (Contact forces)			Kinematics (Foot positions)		
Method	**Mean GRF**	**Max GRF**	**Ballistic GRF**	**Floating**	**Penetration**	**Skate**
MTC [14]	142.7%	9036.7%	120.8%	19.1%	10.0%	16.5%
Kinematics (ours)	119.7%	1252.4%	103.6%	**1.5%**	1.8%	**1.3%**
Physics (ours)	**98.8%**	**293.2%**	**0.0%**	5.9%	**0.1%**	3.8%

Table 3: Physical plausibility evaluation using the *estimated floor* on the synthetic test set. Supplements Table 2 of the main paper.

with the precision, recall, and F1 score of each method. These give additional insight compared to accuracy since data labels are slightly imbalanced (more in-contact frames than no-contact).

Table 1 shows an ablation study between different input and output window size combinations for our network. *Input Window* is the number of frames of 2D lower-body joints given to the network, and *Prediction Window* is the number of frames for which the network outputs foot contact classifications. We use an input window $w = 9$ and prediction window of 5 in our experiments since it achieves the best accuracy on real videos as shown in the table. In general, there is not a clear trend in *Prediction Window* size, but as the *Input Window* size increases, so does accuracy on the real dataset.

Figure 3 shows the accuracy of contact estimations over the entire prediction window of 5 frames on the synthetic test set. Though the target frame in this case is frame index 2, predictions on the off-target frames degrade only slightly and are still very accurate since the input windows is 9 frames. This motivates the use of the majority voting scheme at inference time.

Method	Feet	Body	Body-Align 1
MTC [14]	585.303	**565.068**	**277.296**
Kinematics (ours)	582.400	565.097	281.416
Physics (ours)	**582.311**	587.627	319.517

Table 4: Pose evaluation on synthetic test set using the *estimated floor*. Supplements Table 3 in the main paper.

6.2 Qualitative Motion Evaluation

For extensive qualitative evaluation, see the supplementary video. For evaluating on real data, we use videos from a number of sources: publicly available datasets [2, 11], YouTube videos that are licensed under Creative Commons or with permission from the content creators to be used in this publication, and licensed stock footage. We thank the following YouTube channels that contributed video data: Dance FreaX, Dancercise Studio, Fencer's Edge, MihranTV, DANCE TUTORIALS, Deepak Tulsyan, Gibson Moraes, and pigmie.

6.3 Quantitative Motion Evaluation

Primary quantitative results for motion reconstruction are presented in Section 4.3 of the main paper. As discussed, these quantitative evaluations make use of the ground truth floor plane as input. Note that our method does not *need* the ground truth floor: our floor fitting procedure works well as demonstrated in all qualitative results on live action monocular videos. However, we performed quantitative evaluations on data that contains

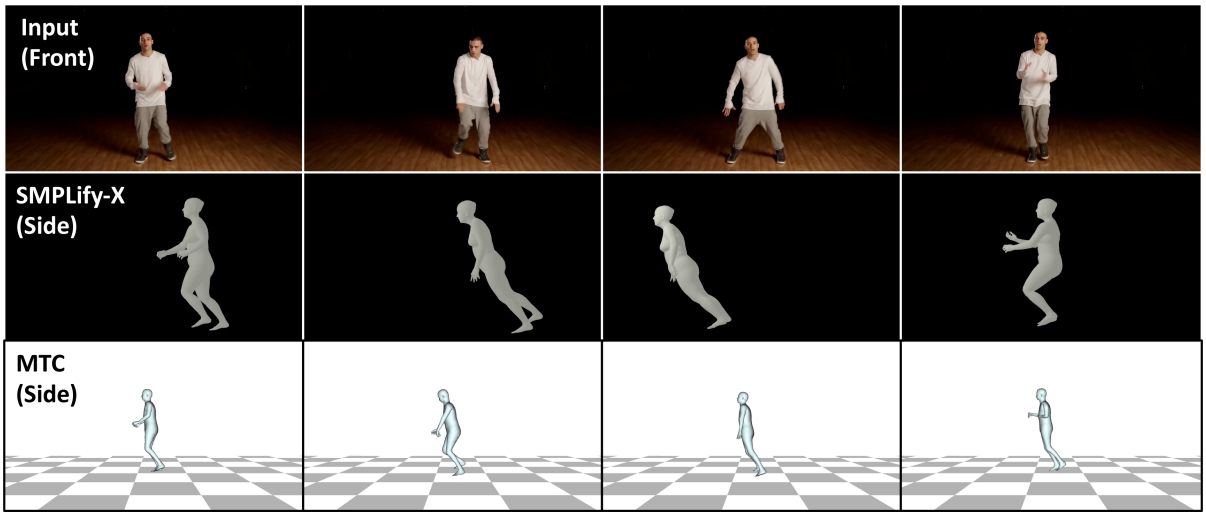

Figure 4: Results from global motion estimation methods on an input dance video. A fixed side view is shown from SMPLify-X [9] and Monocular Total Capture (MTC) [14]. SMPLify-X gives noisy and inconsistent global motion whereas the tracking refinement of MTC gives smoother results. This motivates our decision to use MTC as input to our method and as a baseline in evaluations.

many cases of movement directly towards or away from the camera: a challenging case for MTC, which results in noisy feet joints as input to our method causing a poor floor fit. This makes optimization difficult and interferes with evaluating our primary contributions.

However, for completeness, here we include quantitative results using the floor fitting procedure (rather than taking the ground truth floor as input) on the synthetic test set. Table 3 shows kinematic and dynamics evaluations using the fitted floor while Table 4 shows the pose evaluation. Trends are similar to those seen in the main paper with the ground truth floor.

6.4 Discussion on Global Pose Estimation

For quantitative evaluations in Section 4.3 of the main paper, we do not compare to other methods in global human motion estimation but instead evaluate ablations of our own method: the kinematics-only version and initialization from MTC. The problem of predicting a temporally-consistent global motion (like MTC and this work does) is vastly underexplored so there are few comparable prior works. Many methods do traditional local 3D pose estimation or even predict the global root translation from the camera, but these rarely result in a coherent global human motion.

For example, a recent work from Pavlakos et al. [9] called SMPLify-X estimates global camera extrinsics along with local pose (and body shape) which gives global motion when applied to video. However, we found that MTC, which uses a temporal tracking procedure, gave better results which motivated its use in initializing our pipeline. Figure 4 shows a fixed side view of results from SMPLify-X and MTC on the same video clip. SMPLify-X is noisy and inconsistent especially in terms of global translation; MTC is much smoother and coherent.

6.5 Pose Estimation Evaluation Details

We quantitatively evaluate pose estimation in Section 4.3 of the main paper. We evaluate on our synthetic test set and HumanEva-I [7] walking sequences. Like many pose estimation benchmarks (*i.e.,* Human3.6M [5]), few motions in HumanEva are dynamic with interesting foot contact patterns. Therefore, we evaluate on a subset containing the walking sequences which meet this criteria.

Figure 5: Our method can be applied to multiple characters with varying body masses and mass distributions. From left to right the animated characters are Ybot (body mass 73 kg), Ty (36.5 kg), and Skeleton Zombie (146 kg).

For the MTC baseline, we measure accuracy based directly on the regressed joints given as input to our method. For our method, we use the estimated joints after the full physics-based motion pipeline on our custom skeleton that is initially fit from the MTC input as described in Section 3.1.

For the synthetic test set, we measure joint errors with respect to a subset of the known character rig that includes 16 joints: neck, shoulders, elbows, wrists, hips, knees, ankles, and toes (no spine joints). The "Feet" column of Table 3 in the main paper includes ankle and toe joints only.

On the right side of Table 3 in the main paper, we evaluate methods on the walking sequences from the training split of HumanEva-I [7] (which includes subjects 1, 2, and 3). Following prior work [10], we first split the walking sequences into contiguous chunks by removing corrupted motion capture frames. We then further split these chunks into sequences of roughly 120 frames (about 2 seconds) to use as input to our method. We extract the ground truth floor plane using the camera extrinsics from the dataset and use this as input to our method. Joint errors are measured with respect to an adapted 15-joint skeleton [10] from the HumanEva ground truth motion capture which includes: root, head, neck, shoulders, elbows, wrists, hips, knees, and ankles. The "Feet" column of Table 3 in the main paper includes ankle joints only.

6.6 Multi-Character Generalization

Following the procedure laid out in Section 5, our physics-based optimization can be applied to many character skeletons with varying body and mass distributions. Figure 5 shows an example of estimating motion from the same video for three different characters: Ybot, Ty, and Skeleton Zombie. Ybot has a body mass of 73 kg with a typical human mass distribution [4]. Ty is much lighter at 36.5 kg and his distribution is modified such that 40% of his mass is in the head. Skeleton Zombie is much more massive at 146 kg and has 36% of its mass in its arms alone (due to the giant claws). Our physics-based optimization can handle these variations and still accurately recover the motion from the video. Please see the supplementary video for additional examples.

References

[1] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. In *arXiv preprint arXiv:1812.08008*, 2018.

[2] Caroline Chan, Shiry Ginosar, Tinghui Zhou, and Alexei A Efros. Everybody dance now. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[3] Kwang-Jin Choi and Hyeong-Seok Ko. On-line motion retargetting. In *Proceedings of the 7th Pacific Conference on Computer Graphics and Applications*, PG '99, pages 32–, Washington, DC, USA, 1999. IEEE Computer Society.

[4] Paolo de Leva. Adjustments to zatsiorsky-seluyanov's segment inertia parameters. *Journal of Biomechanics*, 29(9):1223 – 1230, 1996.

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

[6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[7] A. Balan L. Sigal and M. J. Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal of Computer Vision*, 87(1-2), 2010.

[8] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NeurIPS Autodiff Workshop*, 2017.

[9] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3d hands, face, and body from a single image. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[10] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[11] Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, and Sergey Levine. Sfv: Reinforcement learning of physical skills from videos. *ACM Trans. Graph.*, 37(6):178:1–178:14, Dec. 2018.

[12] Zoran Popović and Andrew Witkin. Physically based motion transformation. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 11–20, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[13] Alexander W Winkler, Dario C Bellicoso, Marco Hutter, and Jonas Buchli. Gait and trajectory optimization for legged systems through phase-based end-effector parameterization. *IEEE Robotics and Automation Letters (RA-L)*, 3:1560–1567, July 2018.

[14] Donglai Xiang, Hanbyul Joo, and Yaser Sheikh. Monocular total capture: Posing face, body, and hands in the wild. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.