# Human Dynamics from Monocular Video with Dynamic Camera Movements

RI YU, Seoul National University, South Korea
HWANGPIL PARK, Seoul National University, South Korea
JEHEE LEE, Seoul National University, South Korea



Fig. 1. We reconstructed and simulated a sequence of parkour actions from the dynamic-view video, in which the camera tracks the moving subject.

We propose a new method that reconstructs 3D human motion from in-the-wild video by making full use of prior knowledge on the laws of physics. Previous studies focus on reconstructing joint angles and positions in the body local coordinate frame. Body translations and rotations in the global reference frame are partially reconstructed only when the video has a static camera view. We are interested in overcoming this static view limitation to deal with dynamic view videos. The camera may pan, tilt, and zoom to track the moving subject. Since we do not assume any limitations on camera movements, body translations and rotations from the video do not correspond to absolute positions in the reference frame. The key technical challenge is inferring body translations and rotations from a sequence of 3D full-body poses, assuming the absence of root motion. This inference is possible because human motion obeys the law of physics. Our reconstruction algorithm produces a control policy that simulates 3D human motion imitating the one in the video. Our algorithm is particularly useful for reconstructing highly dynamic movements, such as sports, dance, gymnastics, and parkour actions.

CCS Concepts: • **Computing methodologies** → **Physical simulation**; *Reinforcement learning*; *Motion processing*.

Additional Key Words and Phrases: physics-based simulation, video processing, motion reconstruction

Authors' addresses: Ri Yu, Seoul National University, 1, Gwanak-ro, Gwanak-gu, Seoul, 08826, South Korea, yuri@mrl.snu.ac.kr; Hwangpil Park, Seoul National University, 1, Gwanak-ro, Gwanak-gu, Seoul, 08826, South Korea, hwangppark@gmail.com; Jehee Lee, Seoul National University, 1, Gwanak-ro, Gwanak-gu, Seoul, 08826, South Korea, jehee@mrl.snu.ac.kr.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
0730-0301/2021/12-ART208 $15.00
https://doi.org/10.1145/3478513.3480504

**ACM Reference Format:**
Ri Yu, Hwangpil Park, and Jehee Lee. 2021. Human Dynamics from Monocular Video with Dynamic Camera Movements. *ACM Trans. Graph.* 40, 6, Article 208 (December 2021), 14 pages. https://doi.org/10.1145/3478513.3480504

## 1 INTRODUCTION

Video is an easily accessible source for learning human motor skills. In computer graphics and vision, many researchers have endeavored to estimate human poses and reconstruct 3D human motion from video. Despite the inherent limitation of 2D video (depth ambiguities and occlusion), the quality of 3D poses estimated using deep convolutional networks trained with extensive training datasets has been significantly improved [Kanazawa et al. 2018, 2019; Kocabas et al. 2020; Pavllo et al. 2019].

Current state-of-the-art algorithms often reconstruct a sequence of 3D human poses without body translation and rotation in the global reference frame. Body translation and rotation are estimated only when the video has a static camera view. We are interested in overcoming this static view limitation to deal with dynamic-view videos. The camera may pan, tilt, and zoom to track the moving subject. Body translations and rotations estimated from dynamic-view videos do not correspond to absolute positions in the reference frame. The key technical challenge is inferring body translations and rotations from a sequence of 3D human poses under the assumption that the root motion is unavailable. This inference is possible because human motion obeys the laws of physics.

In this paper, we present a new framework that can reconstruct 3D human motion from in-the-wild video making full use of physics prior based on deep reinforcement learning (DRL). Our algorithm learns a control policy (a.k.a. physics-based controller) that generates 3D human motion that matches the one in the video, in the light of the hints about interactions between the character and the environment given by the user. This indirect reconstruction has several advantages. The reconstructed 3D motion is always physically

plausible since the motion is generated through physics simulation. The learned policy can fill in missing information, such as the lack of root motion, unknown scene arrangement, and imperfect pose estimation.

We are particularly interested in highly dynamic human motion, including sport, dance, and parkour movement. The subject in the video may run through, grab, step on, and jump over objects while the camera tracks the subject. Our algorithm reconstructs both plausible scene arrangement and consistent human movement in the scene simultaneously. Specifically, our main contribution includes:

- We build a new system architecture that combines a 2D pose estimator, a 3D pose estimator, and a policy learning algorithm to reconstruct 3D human motion and scene arrangements simultaneously from dynamic-view video.
- We present a sliding-window algorithm that handles a series of human-object interactions robustly and efficiently while maintaining the global consistency in scene arrangement.
- The contact between the human body and objects is the most important visual cue to understand 3D human motion. We train a deep contact estimator that detects foot-object contacts in dynamic-view video.

## 2 RELATED WORK

### 2.1 Monocular Pose/Motion Estimation

In computer vision, pose estimation techniques from monocular video have remarkably advanced with the use of Convolutional Neural Networks (CNNs), especially ResNet [He et al. 2016]. Toshev et al. [2014] used a cascade of CNNs to regress the coordinates of each joint and estimate a pose in a holistic manner, instead of using the traditional part-based model approaches. Tompson et al. [2014] jointly train a CNN and a graphical model that output a heat-map. In the follow-up studies, different network architectures were explored to enhance the performance of pose estimation [Chen et al. 2018; He et al. 2017; Newell et al. 2016; Pfister et al. 2015; Wei et al. 2016]. Cao et al. [2019] used a bottom-up approach, which detects each joint first and then finds a skeleton, to estimate multi-person poses in real-time.

3D pose estimation has been studied based on the 2D poses estimation techniques described above. Mehta et al. [2017] proposed a real-time, CNN-based framework which regresses 2D and 3D poses on a per-frame basis and generates globally stable 3D poses through skeleton fitting. Pavllo et al. [2019] estimated smooth poses from monocular video using temporal convolution networks. Monszpart et al. [2019] presented a framework to recover human motions from a monocular video and infer scene placements by detecting human-object interactions. As large datasets [Ionescu et al. 2013; Sigal et al. 2010] consisting of 3D motion capture data along with 2D videos are available, 3D pose estimation can be formulated as a standard supervised learning problem without intermediate 2D joint estimation steps. Recently, end-to-end 3D pose estimation approaches which directly infer a 3D pose from a single image or a video were studied. Kanazawa et al. [2018] estimated a 3D pose and a body shape simultaneously from a single image by adopting an adversarial prior which distinguishes the estimated result from real human data. In the follow-up study, Kanazawa et al. [2019] trained a temporal encoder to predict nearby poses given a sequence of images. Kocabas et al. [2020] developed a VIBE framework, which also leverages adversarial learning to construct a motion discriminator from large motion datasets [Mahmood et al. 2019]. Meanwhile, Kolotouros et al. [2019] proposed a collaborated method between a regression-based and an iterative optimization-based approaches.

Optical motion capture has long been considered as the best way to acquire high-quality human motion data, although optical capture systems are costly and cumbersome to setup. Video-based motion capture is emerging as an easily accessible, low-cost, attachment-free alternative. Xu et al. [2018] used CNNs to estimate human performance and non-rigid surface deformation in everyday clothing. A series of studies [Habermann et al. 2019, 2020; Xu et al. 2018] commonly employed template meshes to capture surface details as well as human performance. Xiang et al. [2019] captured 3D body poses as well as facial expression and hand gesture. Shi et al. [2020] focused on maintaining skeletal consistency over time. Most of previous studies assumed that the video is a static shot with a fixed camera setting [Habermann et al. 2019; Mehta et al. 2020; Shi et al. 2020; Xiang et al. 2019; Xu et al. 2018], camera parameters are known [Vondrak et al. 2012], or camera calibration is required [Wei and Chai 2010].

### 2.2 Physics-Based Simulation and Control

Physics-based character simulation and control have been a long-standing challenge in computer graphics. Early approaches relied on state machines and feedback rules [Hodgins et al. 1995; Yin et al. 2007], data-driven tracking mechanisms [da Silva et al. 2008; Lee et al. 2010; Sok et al. 2007], and model-based optimization [Kwon and Hodgins 2017; Liu et al. 2012]. Recently, deep reinforcement learning (DRL) methods have been successfully adopted to the control of physically-simulated articulated characters. The representation power of deep networks achieved accurate control in high-dimensional continuous state/action spaces and successfully simulated various human actions, such as locomotion [Schulman et al. 2015; Yu et al. 2018], stepping stones [Xie et al. 2020], basketball dribbling [Liu and Hodgins 2018], putting on clothes [Clegg et al. 2018], carrying objects [Merel et al. 2020], and even ballet dance [Yuan and Kitani 2020]. Peng et al. [2018a] learned a control policy that allows a simulated character to mimic realistic motion capture data. This imitation learning algorithm has further been improved to deal with interactive control [Bergamin et al. 2019; Park et al. 2019], body shape variations [Won and Lee 2019], and diverse behaviors [Peng et al. 2019; Won et al. 2020].

### 2.3 Motion Reconstruction Leveraging Physics

Motion reconstruction from monocular video is an ill-posed problem. Leveraging law of physics can be an effective way to resolve ambiguity inherent in the problem.

In computer vision and computer graphics fields, many researchers have employed physics-based approaches to resolve the issues such as foot skating, occlusion, and implausible motion when human motions are reconstructed from images or videos. Vondrak et al. [2008] used a physics simulation as a temporal prior in Bayesian filtering for human motion tracking. A series of studies leveraged physics for

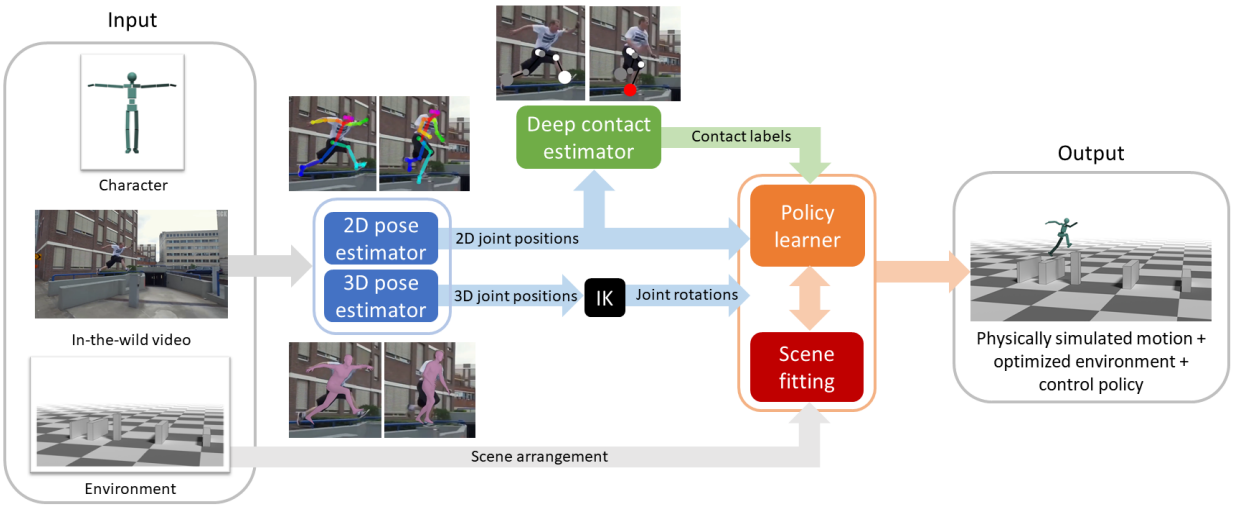



Fig. 2. System overview.

human motion tracking and contact estimation [Brubaker and Fleet 2008; Brubaker et al. 2007, 2009]. Wei et al. [2010] optimized 3D poses at keyframes and generated in-between poses by interpolation. They used physics-based interpolation to eliminate ambiguity caused by occlusion. Rempe et al. [2020] exploited physics-based motion optimization to prevent reconstruction artifacts, such as foot-ground penetration and unbalanced body leaning. They also trained a network to identify foot-ground contact timing from video. Shimada et al. [2020] developed a similar system comprised of a foot contact predictor, 2D and 3D pose estimators, and a physics-based motion optimizer.

Our approach falls into another category of reconstruction methods that estimate control policies generating desired human motion through forward dynamics simulation. Vondrak et al. [2012] used a silhouette likelihood to optimize a physics-based controller. They demonstrated the robustness of their controller on slopes and uneven terrain by changing environment conditions. Yuan and Kitani [2018; 2019] proposed pose estimation methods that reconstruct physically plausible 3D human motion from videos filmed with a head-mounted wearable camera. They used generative adversarial imitation learning [Yuan and Kitani 2018] and reinforcement learning [Yuan and Kitani 2019] for recovering control policies. Recently, they proposed residual force control to mitigate the dynamics mismatch between the simulated character and the real human [Yuan and Kitani 2020], and demonstrated agile human motions such as ballet dance moves. Peng et al. [2018b] combined 2D/3D pose estimators and a policy learner to reproduce dynamic motor skills from 2D video. They successfully reconstructed highly dynamic human motions, such as backflip and cartwheel, assuming empty environment and/or static camera views. Our system is built on top of these prior studies and designed to deal with two technical challenges, dynamic camera viewing and scene arrangements, which have not been addressed yet.

## 3 OVERVIEW

The overview of our system is shown in Figure 2. Our system takes a video clip and a simulated character model as input. We also provide an environment model consisting of a set of simple geometric primitives, such as boxes. The initial size and location of each primitive are given by the user. The scene geometry arrangement is estimated while reconstructing 3D human motion interacting with the geometry primitives.

Our system consists of a 2D pose estimator, a 3D pose estimator, a contact estimator, a policy learner, and a scene geometry builder. The learned pose estimators predict joint positions in 2D screen coordinate and 3D body coordinate systems. The contact estimator predicts when and where body-object contacts occur. Our deep policy learner takes all joints and contact information to learn motor skills and scene arrangement in the global reference coordinate system. The global position and orientation of the skeletal root are automatically recovered while learning motor skills.

## 4 POSE AND CONTACT ESTIMATION

We employed pre-trained 2D/3D pose estimators: The OpenPose framework [Cao et al. 2019] for predicting 2D poses and VIBE [Kocabas et al. 2020] for reconstructing 3D poses from video frames. These pose estimators are trained using large training datasets and their performance depends on the size and distribution of the datasets. Highly-dynamic human motions we wish to reconstruct from the video exhibit challenging poses that deviate significantly from the distribution of common poses in the datasets. Pose estimation sometimes fails when the target pose is leaning or upside-down. To address this issue, we run the 2D pose estimator over 0°, 90°, 180°, and 270° rotated versions of each video frame to select the pose output, as done in previous work [Peng et al. 2018b]. We choose the best result based on temporal coherence and the confidence value of 2D joint estimation. When 3D pose estimation fails, the missing poses are filled in by performing linear interpolation. Estimated 3D

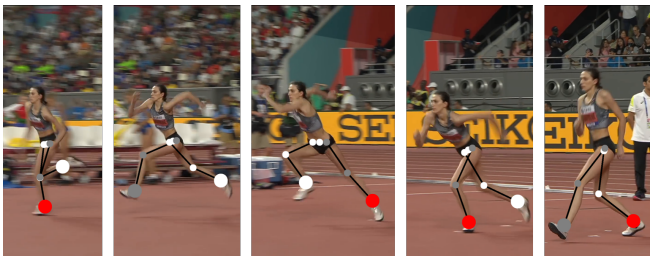


Fig. 3. Contact estimation from the dynamic-view video. The red dots indicate body-ground contact.

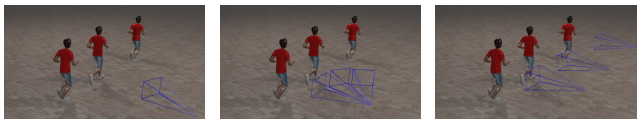


Fig. 4. Three camera settings to synthesize training data for our contact estimator. (Left) a fixed-view camera. (Middle) a panning camera at fixed location. (Right) a tracking camera.

joint positions are converted into an articulated body representation suitable for physics simulation by solving inverse kinematics.

Contact is essential information that represents the physical interaction between the character and its environments including the ground, the wall, and the handrail. Recently, Rempe et al. [2020] trained a network that detects the foot-ground contact from monocular video assuming static camera view. We revised their work to deal with dynamic-view video (see Figure 3). We used 72 motion clips and 11 rigged characters collected from Mixamo motion datasets [Adobe Systems Inc. 2018] to render training images from three camera settings with 31 background textures and 4 light intensities (see Figure 4 and Figure 5). The camera settings include a fixed-view camera, a panning camera tracking the subject at fixed location, and a tracking camera moving together with the subject. The camera position is decided randomly relative to the subject. The motion data include highly-dynamic, contact-rich movements, such as backflip, Capoeira kicks, and breakdance. We label the contacts in the motion data automatically by using the height and velocity of the foot. Using these rendered images as training data, we train a network estimating the contact in a supervised-manner. The contact estimator is augmented to the OpenPose framework. The input video clip is fed into OpenPose to estimate a 2D pose, which consists of nodal points and links between nodes. We use the results of OpenPose as an input to our contact estimator. To take into account the temporal information, the contact estimator observes nine consecutive frames with the target frame in the middle. The contact estimator labels a binary value (1 if in contact) at the each foot node. Our deep contact estimator can provide foot-object contact, and the hand-object contact information is manually given. For more implementation details, please refer to the paper [Rempe et al. 2020].

## 5 LEARNING HUMAN DYNAMICS

The dynamics model of our humanoid character is an articulated figure of rigid links connected by either ball-and-socket or hinge joints.



Fig. 5. Examples of rendered training images for the contact estimator. The motion data used for rendering includes various movements; from simple movements such as walking, running, and push-ups to dynamic movements such as backflip, capoeira, and cartwheel.

The joints are actuated by Proportional-Derivative (PD) servos. The actor in the video physically interacts with objects in the scene. We provide a rough initial arrangement of the scene consisting of simple geometric primitives, such as boxes.

Our goal is to learn a control policy (a.k.a. controller) that operates the humanoid character to mimic 3D human movements in the video based on estimated 2D and 3D poses, contact labels, and body orientation hints computed as explained in the previous section. Let $\pi(\mathbf{a}_t|\mathbf{s}_t, \mathbf{e})$ be the control policy where $\mathbf{s}_t$ and $\mathbf{a}_t$, respectively, are the state and action of the dynamics model at time $t$. The action provides a target pose $\bar{\mathbf{q}}_t$ for PD control at time $t$:

$$\bar{\mathbf{q}}_t = \hat{\mathbf{q}}_t + \mathbf{a}_t, \tag{1}$$

where $\hat{\mathbf{q}}$ is a reference pose made up of all joint orientations with the axis-angle representation from 3D pose estimation and $\mathbf{a}_t$ is an action vector sampled from the control policy to offset the joint orientations. The state is defined by an aggregated vector $\mathbf{s}_t = [\phi_t, \mathbf{p}_t, \mathbf{R}_t, \mathbf{v}_t, \omega_t]$ that includes time $\phi_t \in [0, 1]$ normalized by clip length, the positions of body links, the orientations represented by unit quaternions of body links, and their linear and angular velocities. All features except $\phi_t$ are represented with respect to the body local coordinate system attached to the skeleton root. We define the environment $\mathbf{e}$ as an aggregated vector consisting of the height, size, and position of the objects. Especially, we parameterize the location for the heading direction and the height of each object, and these scene parameters are explored by the scene fitting process. A parameter domain $[\mathbf{e}_{\min}, \mathbf{e}_{\max}]$ to explore is provided by the user. A wider domain allows more flexibility as the requirements for computational cost increase. In our experiments, each location and height parameter has a domain of [-30cm, 30cm].

In this section, we will begin with a baseline algorithm that learns a single control policy that reconstructs 3D human motion throughout the video. This baseline algorithm is good enough with a few environment objects. However, we need a more efficient algorithm to deal with complex scene arrangement, which will be explained later at the end of this section.

### 5.1 Policy Learning

Given a discrete-time dynamics system, deep reinforcement learning searches for a control policy $\pi$ that maximizes the discounted cumulative rewards $V^{\pi}(\mathbf{s})$. At each time $t$, the agent in state $\mathbf{s}_t$ takes action $\mathbf{a}_t$. The environment responds to the action by making

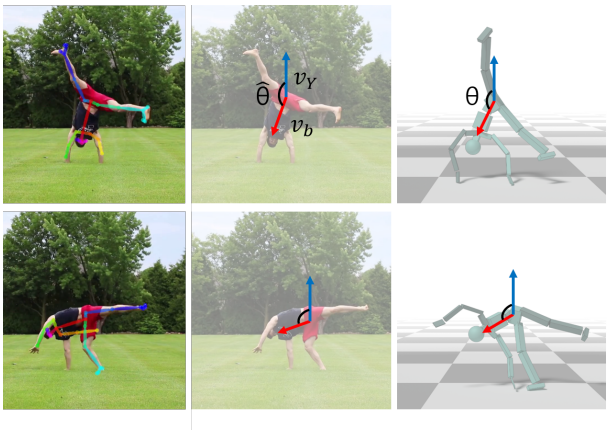



Fig. 6. A body orientation hint from 2D joint positions.

transition to a new state $\mathbf{s}_{t+1}$ and gives a reward $r_t$. The discounted cumulative rewards of policy $\pi$ is defined as follows:

$$V^{\pi}(\mathbf{s}_t) = \mathbb{E}_{\mathbf{s}_0=\mathbf{s}_t, \mathbf{a}_i \sim \pi}\left[\sum_{i=0}^{\infty} \gamma^i r_i\right], \tag{2}$$

where $\gamma \in [0, 1)$ is a discount factor. The optimal control policy $\pi^*$ is

$$\pi^* = \text{argmax}_{\pi} V^{\pi}. \tag{3}$$

The reward design is the key to the success of deep reinforcement learning. Our reward function consists of two terms, $r_{\text{track}}$ and $r_{\text{scene}}$. The former $r_{\text{track}}$ encourages the accurate tracking of estimated poses, while the latter favors good alignment with the scene objects.

$$r = r_{\text{track}} + r_{\text{scene}}. \tag{4}$$

Each of them has several sub-terms. Specifically, $r_{\text{track}}$ includes five sub-terms that evaluate how well the character tracks estimated poses, velocities, body orientation, and contacts with regularization. $r_{\text{scene}}$ includes three sub-terms that govern the alignment and timing of character-object interactions.

$$\begin{aligned} r_{\text{track}} &= w_q r_q + w_v r_v + w_{\text{ori}} r_{\text{ori}} + w_{\text{contact}} r_{\text{contact}} + w_{\text{reg}} r_{\text{reg}}, \\ r_{\text{scene}} &= w_{\text{dist}} r_{\text{dist}} + w_{\text{align}} r_{\text{align}} + w_{\text{com}} r_{\text{com}}. \end{aligned} \tag{5}$$

Individual sub-terms are as follows.

*Pose and Velocity.* $r_q$ and $r_v$ favor a good match between estimated and simulated poses and velocities.

$$\begin{aligned} r_q &= \exp\left(-\alpha_q \left\|\hat{\mathbf{q}} - \mathbf{q}\right\|^2\right), \\ r_v &= \exp\left(-\alpha_v \left\|\hat{\mathbf{v}} - \mathbf{v}\right\|^2\right), \end{aligned} \tag{6}$$

where $\hat{\mathbf{q}}$ is a vector of joint angles of the estimated pose, $\mathbf{q}$ is an aggregated vector of joint angles of the simulated model, and $\mathbf{v}$ is a vector of joint velocities calculated by finite difference. Note that the pose and velocity vectors do not provide any information about the root position and orientation, which are implicitly estimated by deep reinforcement learning.

*Body Orientation.* The body orientation reward favors a good match between body up-vectors estimated from video and simulation. In case of the dynamic-view videos, because both the subject and the camera are dynamic, it is not possible to directly estimate the subject's body position and orientation relative to the reference frame. In cinematography, various camera movements (pan, zoom, dolly, trunk) are allowed, but camera rolling is not advisable. It is strongly recommended to maintain the up-direction steadily in filming. Assuming steady up-direction in the video, we can estimate a valuable hint about the body orientation. The angle $\hat{\theta}$ between a view up-vector $v_Y$ and a body up-vector $v_b$ in the image plane can serve as a hint for the body up-vector in 3D space (see Figure 6). Here, $v_b$ is a 2D vector from the pelvis to the neck obtained from OpenPose result, and $v_Y$ is a 2D unit vector pointing upward in the image plane. We set $\theta$ as the angle between the vector from the position of the pelvis joint to that of the head joint of the character and the y-axis in the 3D space.

$$r_{\text{ori}} = \exp\left(-\alpha_{\text{ori}} \left\|\hat{\theta} - \theta\right\|^2\right) \tag{7}$$

*Contact.* In our implementation, we consider four end-effectors (hands and feet) that may have contact with environment objects. The contact reward favors a good match between estimated contact states $\hat{c}$ and simulated contact states $c$. Here, $c$ and $\hat{c}$ are binary flags.

$$r_{\text{contact}} = \exp\left(-\alpha_{\text{c}} \sum_{l \in \{\text{rf,lf,rh,lh}\}} \left\|\text{xor}(\hat{c}_l, c_l)\right\|^2\right), \tag{8}$$

where the exclusive or operator $\text{xor}(\hat{c}, c) = 1$ if $\hat{c} \neq c$. Otherwise, it is zero.

*Regularization.* The regularization term minimizes joint torques to prevent excessive force and unnecessary movements.

$$r_{\text{reg}} = \exp\left(-\alpha_{\text{reg}} \left\|\tau\right\|^2\right), \tag{9}$$

where $\tau$ is an aggregated vector of joint torques.

*Distance.* The distance reward is intended to ensure that the simulated end-effector is in contact with the desired object when its estimation from the video confirms contact. Let $\mathbf{p}_l$ be the position of the end-effector and $Z_l$ be the target zone on the target object. The target zone is a surface that the subject may step on. Specifically, we set $Z_l$ as an area of a quarter on the center of the surface, colored red in the most left figure in Figure 7.

$$r_{\text{dist}} = \hat{c} \exp\left(-\alpha_{\text{dist}} \sum_{l \in \{\text{rf,lf,rh,lh}\}} \left\|\text{mindist}(Z_l, \mathbf{p}_l)\right\|^2\right), \tag{10}$$

where $\text{mindist}(Z, \mathbf{p})$ is the minimum distance between the area $Z$ and the position $\mathbf{p}$. Note that this reward is non-zero only when the contact flag is on ($\hat{c} = 1$).

We empirically found that, when the contact zones of the hand-contact ($Z_{\text{rh}}$ and $Z_{\text{lh}}$) are the center of the object's upper plane, the policy often failed to reconstruct the motion successfully at the moment of the hand-contact. In this case, we manually set $Z_{\text{rh}}$ and $Z_{\text{lh}}$ to the corner of the object.

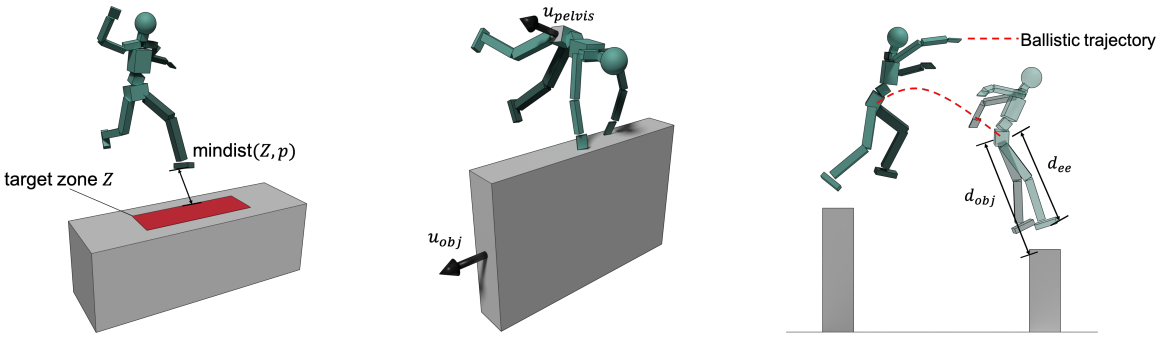



Fig. 7. Scene reward terms. (Left) Distance. (Middle) Alignment. (Right) Center of Mass.

*Alignment.* The character and the object should be well-aligned when they are in contact (see Figure 7(Middle)). We assume that the actor in the video is facing the direction of the object he/she is about to interact with. The alignment reward is

$$r_{\text{align}} = \hat{c} \exp\left(-\alpha_{\text{align}} \left\|1 - \mathbf{u}_{\text{pelvis}} \cdot \mathbf{u}_{\text{obj}}\right\|^2\right), \tag{11}$$

where $\mathbf{u}_{\text{pelvis}}$ and $\mathbf{u}_{\text{obj}}$ are unit vectors represented in the world coordinate system. $\mathbf{u}_{\text{pelvis}}$ is an unit vector along the frontal axis of the pelvis.

*Center of Mass.* When the simulated character is in the air, its center of mass (CoM) follows a ballistic trajectory and its landing timing is determined by the contact estimator. Therefore, the CoM position is predictable. The CoM reward informs the suitability of the ballistic trajectory at an early stage. $d_{\text{ee}}$ is the distance between CoM and the end-effector of reference pose in the landing timing (see Figure 7(Right)). $d_{\text{obj}}$ is the distance between the expected CoM and its expected landing position. If the simulated character perfectly tracks the reference pose and contact timing, $d_{\text{ee}}$ is equal to $d_{\text{obj}}$. The CoM reward is

$$r_{\text{com}} = \exp\left(-\alpha_{\text{com}} \left\|d_{\text{ee}} - d_{\text{obj}}\right\|^2\right), \tag{12}$$

if the character is in the air. Otherwise, the CoM reward is zero.

We used a single set of weights and parameters to generate all examples $w_q = 0.35$, $w_v = 0.1$, $w_{ori} = 0.2$, $w_c = 0.25$, $w_{reg} = 0.1$, $w_{dist} = 0.5$, $w_{align} = 0.2$, $w_{com} = 0.3$, and $\alpha_q = 12$, $\alpha_v = 0.6$, $\alpha_{ori} = 5$, $\alpha_c = 2$, $\alpha_{reg} = 1$, $\alpha_{dist} = 15$, $\alpha_{align} = 1$, and $\alpha_{com} = 5$.

## 5.2 Network Training

The value function $V$ and the control policy $\pi$ are represented by deep neural networks. In the training phase, the experience tuples $(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, r_t)$ are collected episodically through physics simulation, and $V$ and $\pi$ are updated by using Proximal Policy Optimization [Schulman et al. 2017].

As previously discussed by Peng et al. [2018a], dealing with imbalance in the distribution of experience tuples is an important issue in learning trajectory-tracking policies. The simulation system may collect a lot of experience tuples at the beginning of episodes. As the episode proceeds, the simulated character may stumble and fall. The experience tuples after losing balance are useless for learning control policies. We terminate the episodic tracking simulation before it reaches the end of the video if the character stumbles. We monitor the CoM height, collision with obstacles, and the deviation from the estimated poses and body orientation hints to decide if the early termination is needed. This early termination causes sample imbalance issues.

To mitigate the problem, we wish to draw tuples as uniformly as possible in the time horizon. To do so, for each learning iteration, we count how many experience tuples are collected into memory buffer at each frame. The probability distribution $P$ is calculated to be inversely proportional to the number of collected tuples.

$$P(t) = \frac{\frac{1}{n_t+1}}{\sum_i \frac{1}{n_i+1}}, \tag{13}$$

where $n_t$ is the number of tuples sampled at the time $t$. Whenever an episode is completed, a fixed number (32 in our case) of consecutive experience tuples are appended to the memory buffer, and the start time of the appended tuples is selected according to $P$.

## 5.3 Scene Fitting

The computational cost for policy learning depends significantly on the length of the input video and the number of objects in the scene. The algorithm is efficient and reliable for a short video clip recording an action in an empty environment, but the learning process becomes more computationally challenging as the video gets longer and the scene has more objects.

We found that it is not necessary to learn a single policy network that covers the entire time range as a whole. Instead, we learn a sequence of policy networks. Each of the networks learns a control policy in a short time window in which the actor interacts (e.g. contact or jump over) with at most three objects in the scene. We have a series of overlapping time windows that collectively covers the time range (see Figure 8). While learning a policy network in a window, the configuration (e.g., position and height) of the middle object is parameterized with respect to its previous object. By visiting the windows sequentially one-by-one, we can obtain a series of control policies that are robust to the change of environment. We use the $n$-th policy learned with regard to the $n$-th object to control the motion from the moment when the character contacts with $(n-1)$-th object to the moment that the character contacts

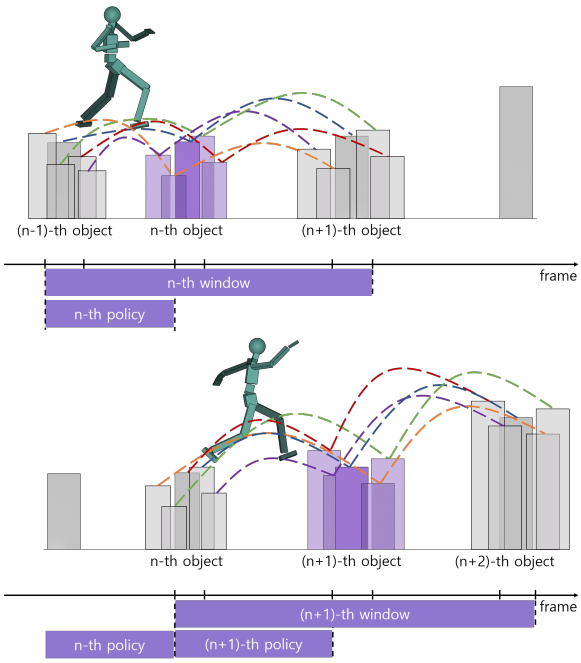



Fig. 8. Learning a sequence of policy networks for sliding time windows.

with $n$-th object. Note that the actor may interact with individual objects multiple times and thus the ordering of the objects may be different from their unique IDs.

We train the policy of each window sequentially. For instance, after training the policy of a window, we train the policy of the next window, where the initial state of this next window is parameterized by the policy trained for the previous window. Once the value and policy networks for each window are learned, the local arrangement of three consecutive objects can be estimated from the networks by maximizing the cumulative value in the window.

The globally-consistent scene arrangement maximizes

$$\mathbf{e}^* = \underset{\mathbf{e}}{\operatorname{argmax}} \sum_{n=1}^{N} f_n(\mathbf{e}), \tag{14}$$

$$f_n(\mathbf{e}) = V^{\pi_n^*}(\mathbf{s}(t_{n-1}), \mathbf{e}) - V^{\pi_n^*}(\mathbf{s}(t_n), \mathbf{e}), \tag{15}$$

where $N$ is the number of windows, $f_n(\mathbf{e})$ is a cumulative reward on the $n$-th window, $\pi_n^*$ is the optimal policy of the $n$-th window, $V^{\pi_n^*}$ is the value function learned along with the policy, and $\mathbf{s}(t_n)$ is the initial state of the $n$-th window. We use Sequential Least Squares Programming (SLSQP) [Kraft 1994] to compute the consistent scene arrangement.

## 6 RESULTS

Our humanoid character is 175 cm tall and weighs 61.4 kg. The skeleton has two hinge joints on the knee, 15 ball-and-socket joints, and the floating base. We used linear-time stable PD [Tan et al. 2011; Yin and Yin 2020] for robust actuation of the linked body dynamics system. The proportional and derivative gains for PD control are $k_\mathrm{p} = 600$ and $k_\mathrm{d} = 49 \approx 2\sqrt{k_\mathrm{p}}$. The DART physics engine is used to simulate the dynamics system [Lee et al. 2018]. The control frequency is equal to the video frame rate, which ranges from 18Hz to 30Hz (see Table 1). The simulation frequency is ten times the control frequency if the simulation involves hand-object interactions, and five times otherwise.

Table 1. Input video details.

| Sequences | No. of frames | fps | Camera types |
|---|---|---|---|
| Cartwheel | 82 | 30 | static |
| Backflip | 67 | 30 | static |
| Forward jump | 75 | 30 | static |
| Dance | 69 | 24 | static |
| Lateral jump | 130 | 24 | dynamic |
| Jump top-out | 86 | 24 | dynamic |
| Hurdle | 17 | 18 | dynamic |
| High jump | 72 | 25 | dynamic |
| Parkour1 | 78 | 25 | dynamic |
| Parkour2 | 87 | 30 | dynamic |
| Parkour3 | 66 | 30 | dynamic |
| Parkour4 | 70 | 30 | dynamic |

We used PyTorch [Paszke et al. 2019] as a neural network library for implementing contact estimation and reinforcement learning. The deep contact estimator has four fully-connected layers which have 1024, 512, 128, 32 nodes, respectively, with batch normalization. The policy and value networks have three fully-connected layers of 128 nodes. We used ReLu activation function and initialized the network parameters using He initialization [He et al. 2015]. The Adam optimizer with learning rate 0.0001 is used to train the contact estimator. The gradient descent with Nesterov momentum is used to train policy and value networks, where its momentum is 0.9 and learning rate is 0.0002. The discount factor and the GAE parameter for deep reinforcement learning are $\gamma = 0.99$ and $\lambda = 0.95$, respectively. The minibatch size for the contact estimator network is 64, and the minibatch size for the policy and value networks is 256. The computation time for deep reinforcement learning varies depending on the length of video clip and the number/type of human-object interactions. It takes 24 hours to 96 hours on a single PC with Intel Core i9-9900K (3.6 GHz, 8 cores). Our code and data is available at https://github.com/yul85/movingcam.

### 6.1 Video Clips

We tested our algorithm with many monocular video clips. Please refer to the accompanied video for the reconstruction results. Most video clips downloaded from YouTube exhibit energetic, highly-dynamic human movements. Some of the video clips are static-shots (Figure 9), while the others have dynamic camera views (Figure 10 and Figure 11). Static-shot clips were taken from the dataset of Peng et al. [2018b]. Detailed information on video clips are summarized in Table 1.

Reconstructing parkour moves demonstrates the full potential of our algorithm. Parkour athletes use dynamic movements, such as running, vaulting, jumping, and rolling, to move from one point to another exploiting environment features. They use both hands and

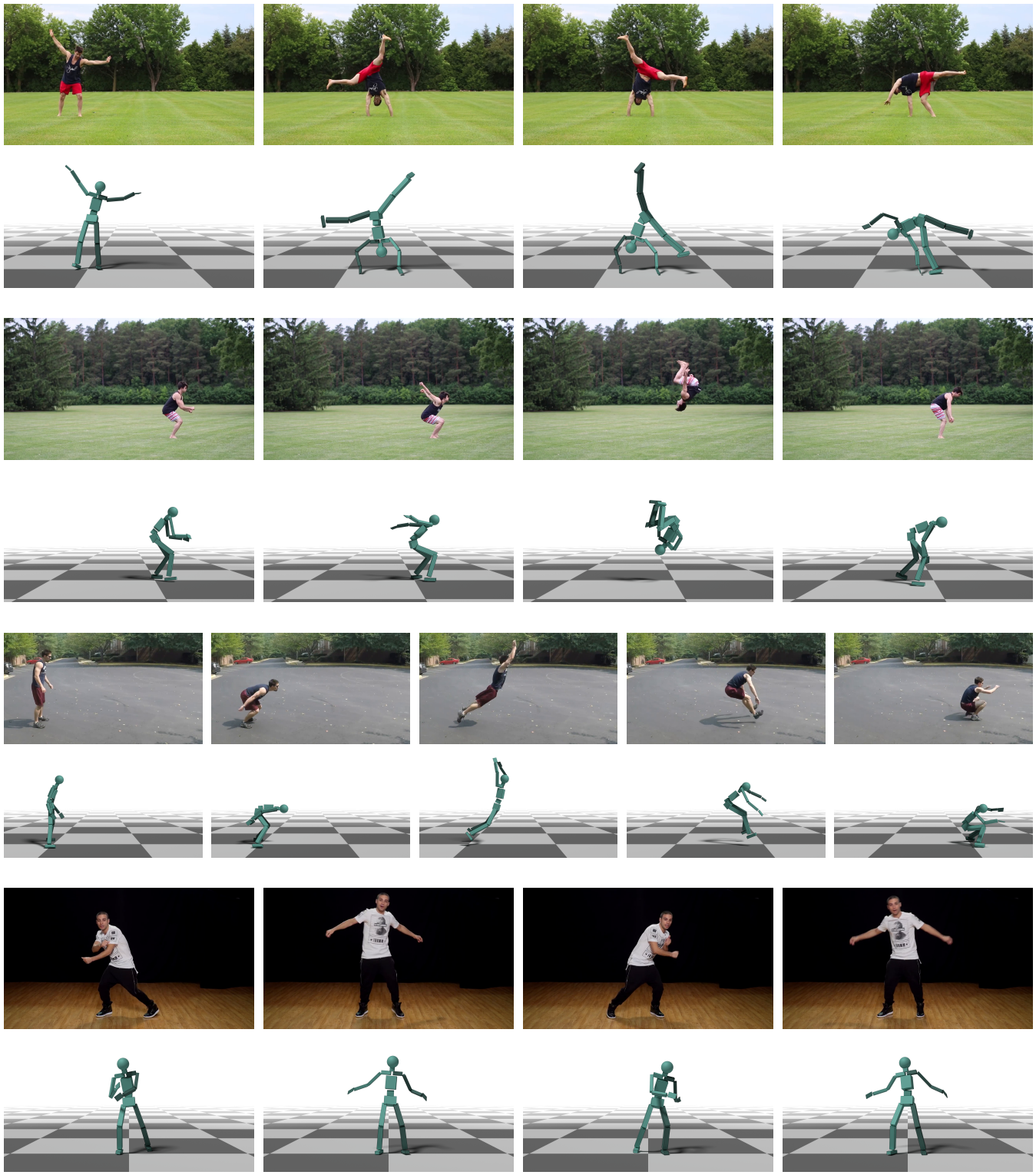


Fig. 9. Motion reconstruction from static-shot video clips. (From top to bottom) Cartwheel, backflip, forward jump, and dance.

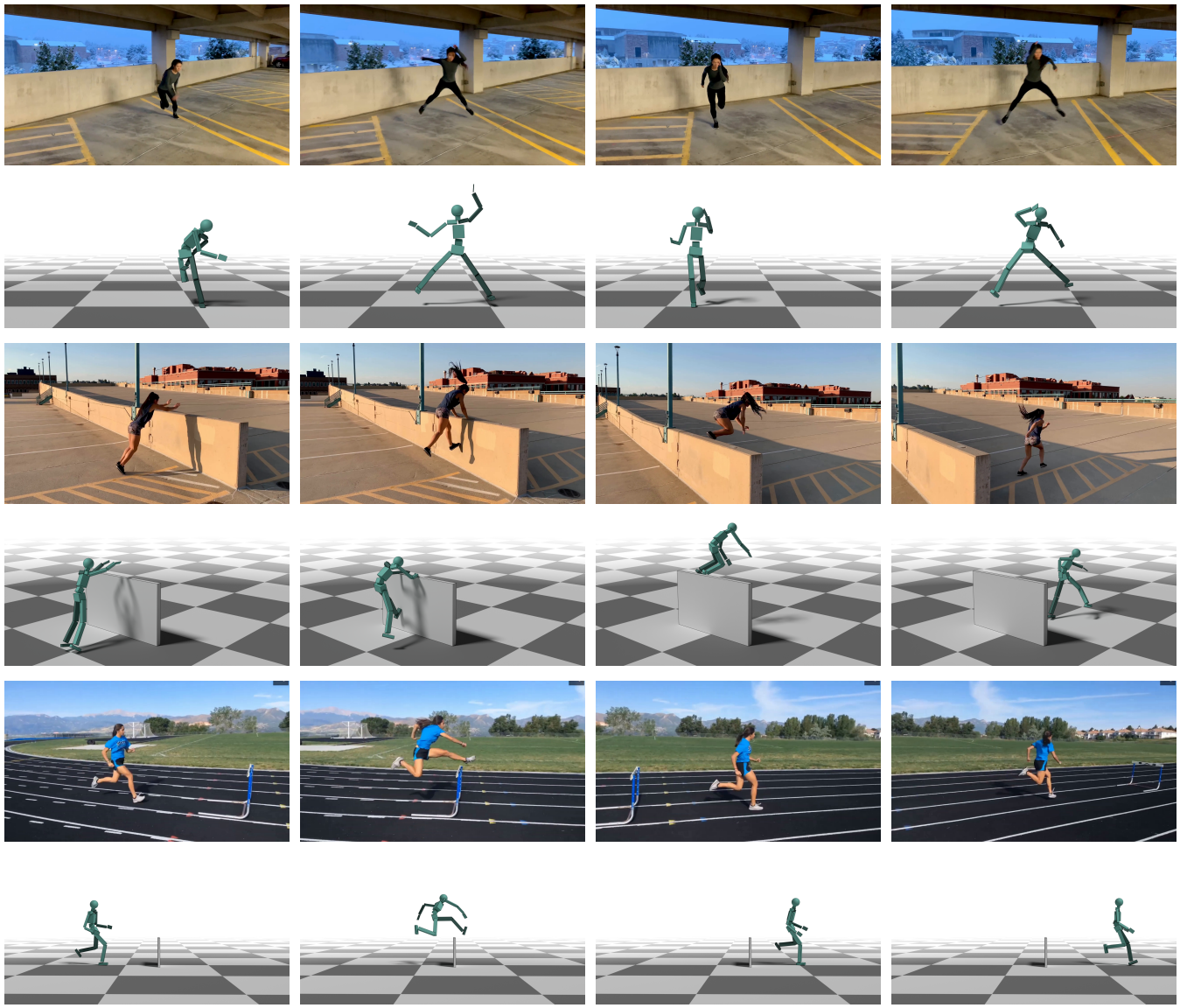


Fig. 10. Motion reconstruction from dynamic-shot video clips. (From top to bottom) Lateral jump, jump top-out, and hurdle.

feet to jump, grasp, kick, and roll. The camera tracks the athletes to record their fluid movements.

The high jump is yet another challenging example because the 3D pose estimator fails to estimate for over 30 consecutive frames during the athlete flies over the bar. Although our algorithm can deal with small gaps gracefully, the 30 frame gap is beyond its capacity. We provided two manually-keyframed 3D poses in the middle of jump. The keyframed poses are assigned with low weights ($w_q = 0.1$) to give them less credibility than automatically estimated poses. Our algorithm successfully reconstructed a smooth, physically-valid 3D motion filling in the large gap with sparse keyframes (see Figure 11 (top)). The high jump is the only example we used manual keyframes. All the other examples are generated with automatic 2D/3D pose estimation.

## 6.2 Effectiveness of Scene Estimation

In our implementation, the user provides a rough initial guess on the scene arrangement, which is refined while learning the value and policy networks. To evaluate the effectiveness of this approach, we compare the reconstruction accuracy of the user's guess and estimated scene arrangement. We used two measures for quantitative comparison: MPJPE and contact accuracy. The mean per joint position error (MPJPE) measures the difference between VIBE estimated

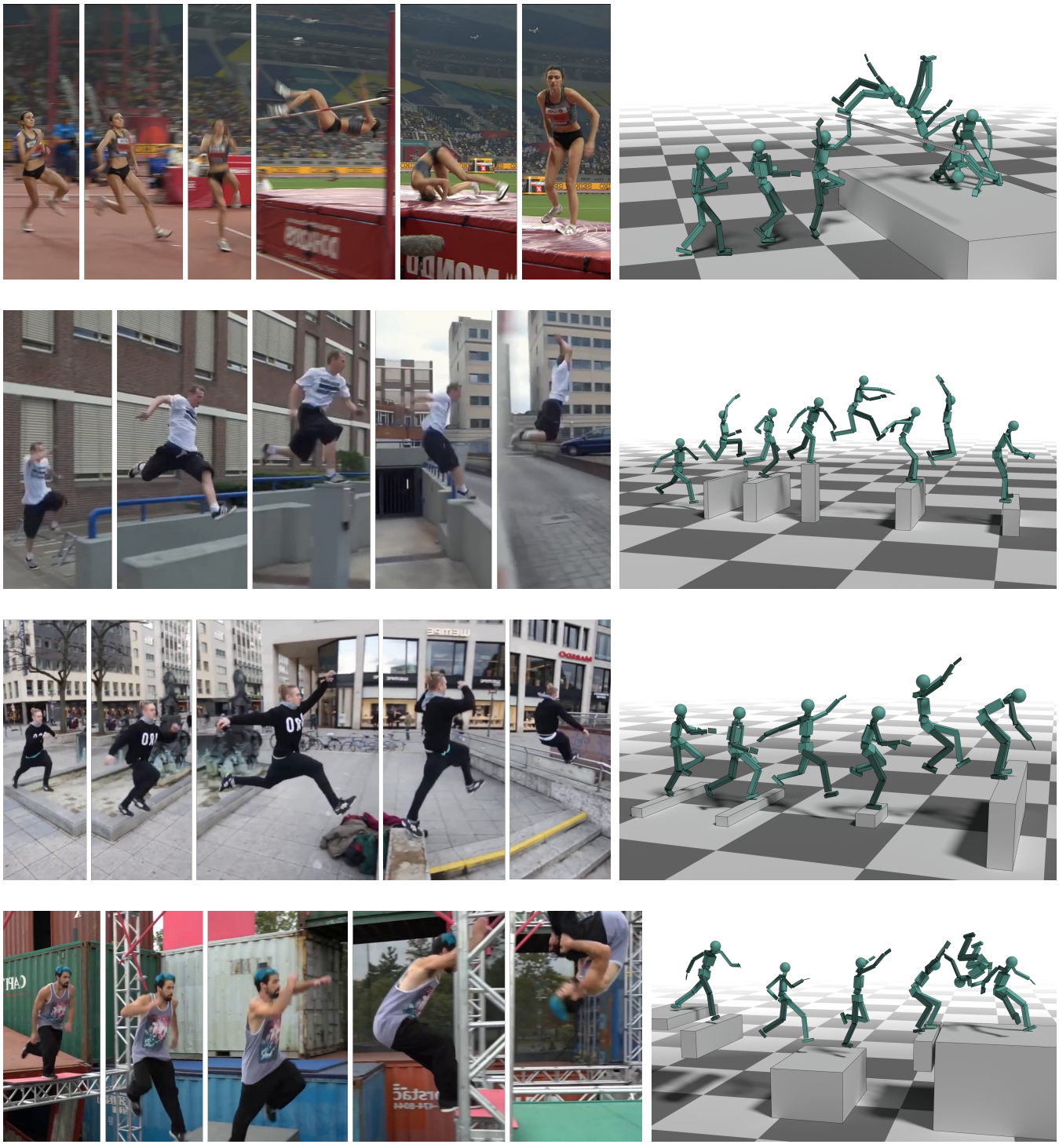


Fig. 11. High jump and parkour sequences.

Table 2. Quantitative comparisons between fixed and estimated scene arrangements.

| | MPJPE [mm] ↓ | | Contact accuracy [%] ↑ | |
|---|---|---|---|---|
| | human Intuition | Object Generator | human Intuition | Object Generator |
| Parkour1 | 137.21 | **134.24** | 81.8 | **85.2** |
| Parkour3 | 181.98 | **142.34** | 50.8 | **93.9** |

poses and simulated poses.

$$\text{MPJPE} = \frac{1}{I}\frac{1}{J}\sum_{i=0}^{I}\sum_{j=0}^{J}\left\|p_j^i - \hat{p}_j^i\right\|, \tag{16}$$

where $I$ is the number of motion frames, $J$ is the number of joints, $\hat{p}_j^i$ is the $j$-th joint position of the reference pose at frame $i$, and $p_j^i$ is the $j$-th joint position of the simulated pose at frame $i$. The contact accuracy measures the average error of contact mismatches between the ground truth and the simulation.

The estimated scene arrangement achieves lower MPJPE and higher contact accuracy than the policy with the fixed scene arrangement provided by the user (see Table 2). Our approach co-estimating the motion and the scene arrangement provides better flexibility and higher accuracy. Even so, the user-provided arrangement is still important in practice because good initial guess helps faster learning and convergence.

*Robustness.* A series of learned control policies convey far more information than motion reconstruction in a fixed scene arrangement. We can interactively generate a large variety of novel scene arrangements, for which the control policies synthesize physically-plausible 3D human movements (see Fig 12). This example shows the potential of a reconstruct-and-generalize framework that unifies the data acquisition process and the data generation process.

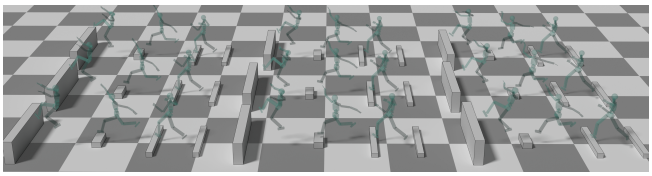


Fig. 12. The learned control policy can robustly generate motions in various scene arrangements.

## 6.3 Comparison of Contact Estimators

We compared the performance of contact estimators (see Table 3). The network of Rempe et al. [2020] was trained with static-view videos, while we trained our network with a mixture of static-view and dynamic-view videos. The accuracy of the foot contact estimation is evaluated for the example videos used in this paper. We manually labelled test images to provide ground truth. As expected, two networks performed comparably with static-view videos since they share the same network architecture. Our network performs better than Rempe's network for all dynamic-view videos by 8.84% in average. This comparison shows that our contact estimator can effectively deal with dynamic-view videos.

## 6.4 Ablation Study

*Reward.* We conducted ablation studies to demonstrate the effectiveness of the rewards we designed. Note that contact reward $r_c$ and body orientation reward $r_{\text{ori}}$ use the hints obtained from the 2D pose estimation results. Also, we trained the policies without the reward terms related to the character-object interaction, which are the distance $r_{dist}$, the alignment $r_{align}$, and the CoM $r_{com}$ rewards.

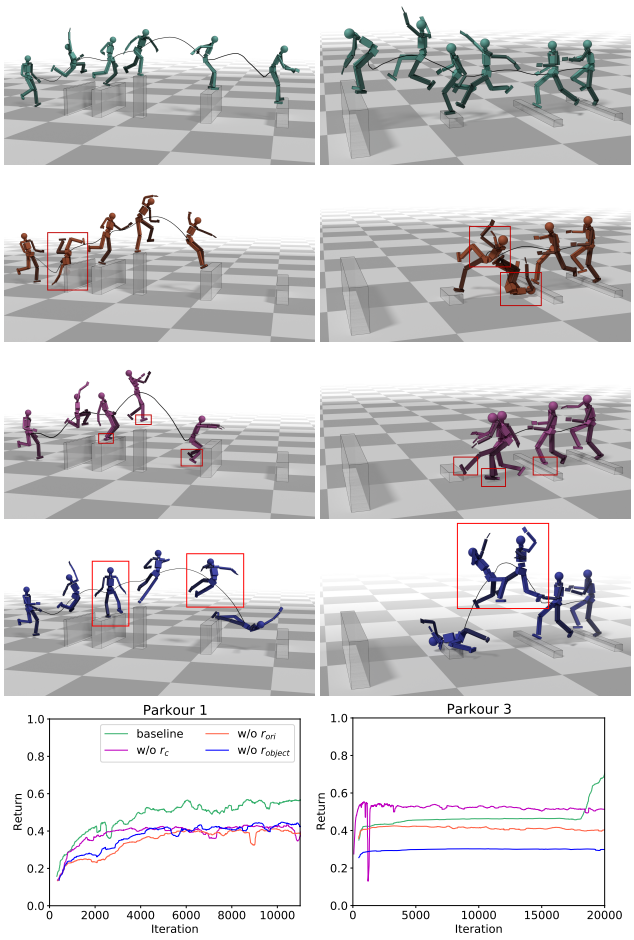



Fig. 13. Ablation study for rewards. (Top) The baseline algorithm (ours) successfully reconstructed the Parkour moves. (Second row) Without body orientation hints, the simulated character fails to maintain the upright direction. (Third row) Without contact rewards, the simulated character fails to land on appropriate location. (Fourth row) Without $r_{\text{object}}$, which indicates the distance, the alignment, and the CoM rewards, the simulated character cannot advance the sequence towards the right direction and fails to land on the object at the expected contact time. (Bottom) Learning curves confirm the benefits of the rewards we designed.

Figure 13 shows the comparison with and without the reward terms. Without the body orientation hints, the simulated character

Table 3. Accuracy (%) of foot contact estimation.

| | Static | | | | Dynamic | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Cartwheel | Backflip | Forward jump | Dance | Lateral jump | Jump top-out | Hurdle | High jump | Parkour1 | Parkour2 | Parkour3 | Parkour4 |
| Ours | 91.7 | 88.1 | **96.5** | 88.5 | **91.1** | **80.8** | **87.0** | **87.4** | **90.0** | **82.3** | **82.3** | **90.8** |
| Rempe et al. | **94.9** | **93.7** | 90.1 | **89.2** | 75.0 | 71.7 | 82.6 | 73.6 | 81.4 | 75.9 | 75.8 | 85.0 |

often falls over since the training data does not provide any information about the body orientation (see the second row in Figure 13). Although the hint estimated from the image is not accurate, it still helps the simulated character correctly estimate its upright direction. Without the contact rewards, the simulated character often fails to land on appropriate contact zone and eventually stumbles (see the third row in Figure 13). Without the distance, the alignment, and the CoM rewards, the simulated character faces the wrong direction, thus cannot proceed with the parkour sequence properly. Moreover, it even spins in the air. Because of these excessive and redundant movements, the character can not reach the object at the expected contact time (see the fourth row in Figure 13). However, our algorithm equipped with all the reward terms successfully reconstructed the parkour moves jumping over multiple obstacles (see the first row in Figure 13).

*Adaptive Sampling.* We also conducted an ablation study to evaluate the effectiveness of adaptive sampling in Equation (13). Figure 14 shows the learning curves with and without adaptive sampling. As expected, adaptive sampling is essential if the input video is long and/or it exhibits complex scene arrangement.

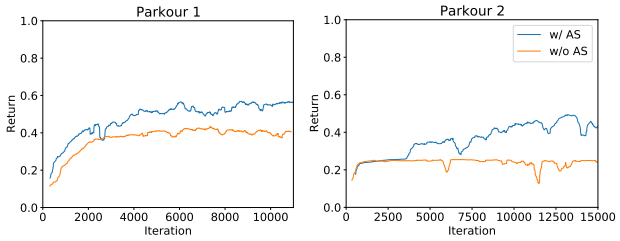
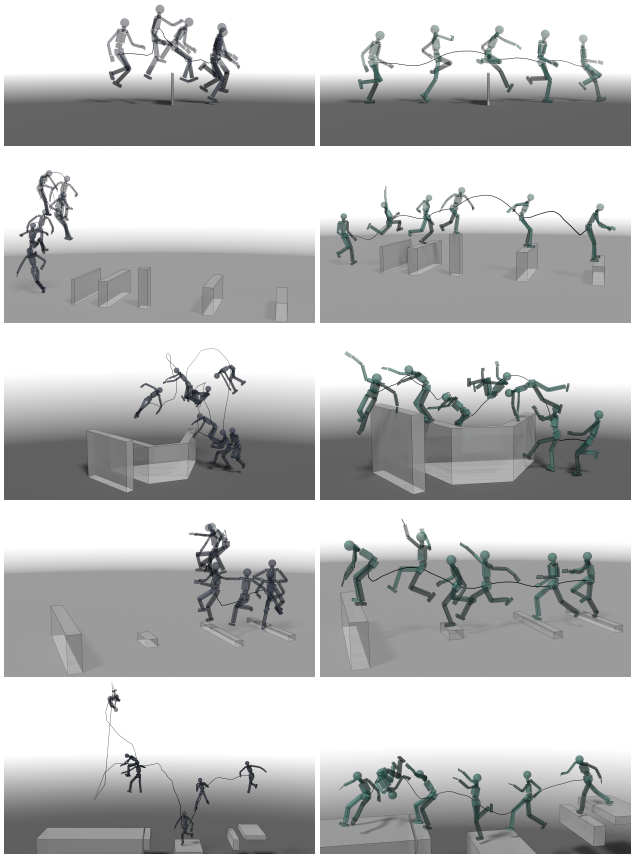



Fig. 14. Learning curves with and without adaptive sampling (AS) for long sequences, parkour1 and parkour2.

## 6.5 Comparison to SFV

Figure 15 shows side-by-side comparisons between SFV [Peng et al. 2018b] and our method. In SFV framework, after the motion reconstruction process, they do an additional post-process to recover the global transformation of the motion. Which is to get the character's trajectory by taking the pixel position of each frame of the character, assuming there's no camera movement. Then, they do train the control policy to mimic the reconstructed motion data as a reference. Therefore, the results reconstructed from dynamic-camera videos using SFV framework do not have coherent global root trajectories. On the other hand, our framework successfully reconstructed motion trajectories regardless of camera movements.



(a) SFV [Peng et al. 2018b] (b) Ours

Fig. 15. Qualitative comparison between SFV [Peng et al. 2018b] and our method. Figures in the left column show reconstructed motions from SFV (dark blue characters), and our motion reconstruction results (green characters) are in the right columns. The global root trajectories are illustrated as black lines.

## 7 DISCUSSION

We present a DRL framework that reconstructs 3D human motion from monocular, dynamic-view video by learning control policy. The learned policy provides us with a deeper understanding of the reconstructed motion's underlying dynamics and opportunities to generalize. The learned policies in our experiments show detailed motor skills of how humans run, jump, change the moving direction rapidly, pull the body above the wall, swing, and flip. Such details are often not obvious from video viewing. Our algorithm unveils

the coordination of human movements and motor skills from in-the-wild videos.

The performance of our algorithm depends on the performance of 2D and 3D pose estimators. As demonstrated in our experiments, our algorithm can deal with moderate gaps caused by pose estimation failure. However, large gaps over many consecutive frames can be problematic. Our framework is agnostic to the choice of 2D and 3D pose estimators. As better pose estimators are developed in the future, the performance of the algorithm will improve accordingly.

Our framework has numerous limitations. We make a number of assumptions to keep the problem manageable. Our dynamics model may not match the body of the subject in the video. Fortunately, the learning algorithm is quite robust to the choice of physics parameters, such as mass, inertia, friction, and restitution coefficients. Our algorithm can currently handle only a few types of human-object interactions that include stepping, grabbing, and kicking. There are many other types we can think of, such as rolling/sliding contact and deformation, which pose interesting challenges in physics simulation. The deep contact estimator works well in detecting foot-ground or foot-object contacts. However, it is not easy to train contact estimators for hands, knees, and elbows, mainly because we do not have sufficiently large training datasets for various contact types. In our experiments, we manually labeled hand-ground and hand-object contacts.

There are exciting opportunities for future work. A big challenge would be the motion reconstruction of multiple humans interacting with each other. Understanding the dynamics of human-human interaction has many applications in activity recognition and synthesis. Currently, we assume that human movement in the video takes place on the surface of the Earth. It would be nice to reconstruct 3D human movements in different physical environments, such as underwater and lunar surface.

## ACKNOWLEDGMENTS

This work was supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No.2017-0-00878, (SW StarLab) Human motion simulation based on deep learning).

## REFERENCES

Adobe Systems Inc. 2018. *Mixamo.* http://www.mixamo.com

Kevin Bergamin, Simon Clavet, Daniel Holden, and James Richard Forbes. 2019. DReCon: data-driven responsive control of physics-based characters. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–11.

Marcus A Brubaker and David J Fleet. 2008. The kneed walker for human pose tracking. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1–8.

Marcus A Brubaker, David J Fleet, and Aaron Hertzmann. 2007. Physics-based person tracking using simplified lower-body dynamics. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1–8.

Marcus A Brubaker, Leonid Sigal, and David J Fleet. 2009. Estimating contact dynamics. In *2009 IEEE 12th International Conference on Computer Vision*. IEEE, 2389–2396.

Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. 2019. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019).

Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. 2018. Cascaded pyramid network for multi-person pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7103–7112.

Alexander Clegg, Wenhao Yu, Jie Tan, C. Karen Liu, and Greg Turk. 2018. Learning to Dress: Synthesizing Human Dressing Motion via Deep Reinforcement Learning. *ACM Trans. Graph.* 37, 6, Article 179 (Dec. 2018), 10 pages.

Marco da Silva, Yeuhi Abe, and Jovan Popović. 2008. Interactive Simulation of Stylized Human Locomotion. *ACM Trans. Graph.* 27, 3 (Aug. 2008), 1–10.

Marc Habermann, Weipeng Xu, Michael Zollhoefer, Gerard Pons-Moll, and Christian Theobalt. 2019. Livecap: Real-time human performance capture from monocular video. *ACM Transactions on Graphics (TOG)* 38, 2 (2019), 1–17.

Marc Habermann, Weipeng Xu, Michael Zollhofer, Gerard Pons-Moll, and Christian Theobalt. 2020. Deepcap: Monocular human performance capture using weak supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5052–5063.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*. 2961–2969.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *CoRR* abs/1502.01852 (2015). arXiv:1502.01852 http://arxiv.org/abs/1502.01852

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

Jessica K. Hodgins, Wayne L. Wooten, David C. Brogan, and James F. O'Brien. 1995. Animating Human Athletics. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '95)*. Association for Computing Machinery, New York, NY, USA, 71–78.

Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. 2013. 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 (2013), 1325–1339.

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

Angjoo Kanazawa, Jason Y. Zhang, Panna Felsen, and Jitendra Malik. 2019. Learning 3D Human Dynamics from Video. In *Computer Vision and Pattern Recognition (CVPR)*.

Muhammed Kocabas, Nikos Athanasiou, and Michael J Black. 2020. VIBE: Video inference for human body pose and shape estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5253–5263.

Nikos Kolotouros, Georgios Pavlakos, Michael J Black, and Kostas Daniilidis. 2019. Learning to reconstruct 3D human pose and shape via model-fitting in the loop. In *Proceedings of the IEEE International Conference on Computer Vision*. 2252–2261.

Dieter Kraft. 1994. Algorithm 733: TOMP–Fortran Modules for Optimal Control Calculations. *ACM Trans. Math. Softw.* 20, 3 (Sept. 1994), 262–281. https://doi.org/10.1145/192115.192124

Taesoo Kwon and Jessica K. Hodgins. 2017. Momentum-Mapped Inverted Pendulum Models for Controlling Dynamic Human Motions. *ACM Trans. Graph.* 36, 4, Article 145d (Jan. 2017), 14 pages.

Jeongseok Lee, Michael X. Grey, Sehoon Ha, Tobias Kunz, Sumit Jain, Yuting Ye, Siddhartha S. Srinivasa, Mike Stilman, and C. Karen Liu. 2018. DART Dynamic Animation and Robotics Toolkit. *Journal of Open Source Software* 3, 22 (2018), 500.

Yoonsang Lee, Sungeun Kim, and Jehee Lee. 2010. Data-Driven Biped Control. *ACM Trans. Graph.* 29, 4, Article 129 (July 2010), 8 pages.

Libin Liu and Jessica Hodgins. 2018. Learning basketball dribbling skills using trajectory optimization and deep reinforcement learning. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–14.

Libin Liu, KangKang Yin, Michiel van de Panne, and Baining Guo. 2012. Terrain Runner: Control, Parameterization, Composition, and Planning for Highly Dynamic Motions. *ACM Trans. Graph.* 31, 6, Article 154 (Nov. 2012), 10 pages.

Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. 2019. AMASS: Archive of motion capture as surface shapes. In *Proceedings of the IEEE International Conference on Computer Vision*. 5442–5451.

Dushyant Mehta, Oleksandr Sotnychenko, Franziska Mueller, Weipeng Xu, Mohamed Elgharib, Pascal Fua, Hans-Peter Seidel, Helge Rhodin, Gerard Pons-Moll, and Christian Theobalt. 2020. XNect: Real-time multi-person 3D motion capture with a single RGB camera. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 82–1.

Dushyant Mehta, Srinath Sridhar, Oleksandr Sotnychenko, Helge Rhodin, Mohammad Shafiei, Hans-Peter Seidel, Weipeng Xu, Dan Casas, and Christian Theobalt. 2017. VNect: Real-time 3D Human Pose Estimation with a Single RGB Camera. *ACM Transactions on Graphics* 36, 4, 14 pages. https://doi.org/10.1145/3072959.3073596

Josh Merel, Saran Tunyasuvunakool, Arun Ahuja, Yuval Tassa, Leonard Hasenclever, Vu Pham, Tom Erez, Greg Wayne, and Nicolas Heess. 2020. Catch & Carry: Reusable Neural Controllers for Vision-Guided Whole-Body Tasks. *ACM Trans. Graph.* 39, 4, Article 39 (July 2020), 14 pages.

Aron Monszpart, Paul Guerrero, Duygu Ceylan, Ersin Yumer, and Niloy J Mitra. 2019. iMapper: Interaction-guided scene mapping from monocular videos. *ACM Transactions On Graphics (TOG)* 38, 4 (2019), 1–15.

Alejandro Newell, Kaiyu Yang, and Jia Deng. 2016. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*. Springer, 483–499.

Soohwan Park, Hoseok Ryu, Seyoung Lee, Sunmin Lee, and Jehee Lee. 2019. Learning predict-and-simulate policies from unorganized human motion data. *ACM*

*Transactions on Graphics (TOG)* 38, 6 (2019), 1–11.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

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

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. 2018a. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–14.

Xue Bin Peng, Michael Chang, Grace Zhang, Pieter Abbeel, and Sergey Levine. 2019. MCP: Learning composable hierarchical control with multiplicative compositional policies. *arXiv preprint arXiv:1905.09808* (2019).

Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, and Sergey Levine. 2018b. SFV: Reinforcement Learning of Physical Skills from Videos. *ACM Trans. Graph.* 37, 6, Article 178 (Nov. 2018), 14 pages.

Tomas Pfister, James Charles, and Andrew Zisserman. 2015. Flowing convnets for human pose estimation in videos. In *Proceedings of the IEEE International Conference on Computer Vision*. 1913–1921.

Davis Rempe, Leonidas J. Guibas, Aaron Hertzmann, Bryan Russell, Ruben Villegas, and Jimei Yang. 2020. Contact and Human Dynamics from Monocular Video. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-Dimensional Continuous Control Using Generalized Advantage Estimation. arXiv:1506.02438 [cs.LG]

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347 [cs.LG]

Mingyi Shi, Kfir Aberman, Andreas Aristidou, Taku Komura, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. 2020. MotioNet: 3D Human Motion Reconstruction from Monocular Video with Skeleton Consistency. *ACM Transactions on Graphics (TOG)* 40, 1 (2020), 1–15.

Soshi Shimada, Vladislav Golyanik, Weipeng Xu, and Christian Theobalt. 2020. PhysCap: Physically Plausible Monocular 3D Motion Capture in Real Time. *ACM Transactions on Graphics* 39, 6, Article 235 (dec 2020).

Leonid Sigal, Alexandru O Balan, and Michael J Black. 2010. 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), 4.

Kwang Won Sok, Manmyung Kim, and Jehee Lee. 2007. Simulating Biped Behaviors from Human Motion Data. *ACM Trans. Graph.* 26, 3 (July 2007), 107:1–107:10.

J. Tan, K. Liu, and G. Turk. 2011. Stable Proportional-Derivative Controllers. *IEEE Computer Graphics and Applications* 31, 4 (2011), 34–44. https://doi.org/10.1109/MCG.2011.30

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

Alexander Toshev and Christian Szegedy. 2014. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1653–1660.

Marek Vondrak, Leonid Sigal, Jessica Hodgins, and Odest Jenkins. 2012. Video-based 3D motion capture through biped control. *ACM Transactions On Graphics (TOG)* 31, 4 (2012), 1–12.

Marek Vondrak, Leonid Sigal, and Odest Chadwicke Jenkins. 2008. Physical simulation for probabilistic motion tracking. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1–8.

Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. 2016. Convolutional pose machines. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 4724–4732.

Xiaolin Wei and Jinxiang Chai. 2010. Videomocap: Modeling physically realistic human motion from monocular video sequences. In *ACM SIGGRAPH 2010 papers*. 1–10.

Jungdam Won, Deepak Gopinath, and Jessica Hodgins. 2020. A Scalable Approach to Control Diverse Behaviors for Physically Simulated Characters. *ACM Trans. Graph.* 39, 4, Article 33 (July 2020), 12 pages.

Jungdam Won and Jehee Lee. 2019. Learning body shape variation in physics-based characters. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–12.

Donglai Xiang, Hanbyul Joo, and Yaser Sheikh. 2019. Monocular total capture: Posing face, body, and hands in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 10965–10974.

Zhaoming Xie, Hung Yu Ling, Nam Hee Kim, and Michiel van de Panne. 2020. ALLSTEPS: Curriculum-driven Learning of Stepping Stone Skills. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 213–224.

Weipeng Xu, Avishek Chatterjee, Michael Zollhöfer, Helge Rhodin, Dushyant Mehta, Hans-Peter Seidel, and Christian Theobalt. 2018. MonoPerfCap: Human Performance Capture From Monocular Video. *ACM Trans. Graph.* 37, 2, Article 27 (May 2018), 15 pages. https://doi.org/10.1145/3181973

KangKang Yin, Kevin Loken, and Michiel van de Panne. 2007. SIMBICON: Simple Biped Locomotion Control. *ACM Trans. Graph.* 26, 3 (July 2007), 105:1–105:10.

Zhiqi Yin and KangKang Yin. 2020. Linear Time Stable PD Controllers for Physics-based Character Animation. *Computer Graphics Forum* 39, 8 (2020), 191–200. https://doi.org/10.1111/cgf.14113

Wenhao Yu, Greg Turk, and C Karen Liu. 2018. Learning symmetric and low-energy locomotion. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–12.

Ye Yuan and Kris Kitani. 2018. 3d ego-pose estimation via imitation learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 735–750.

Ye Yuan and Kris Kitani. 2019. Ego-pose estimation and forecasting as real-time pd control. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 10082–10092.

Ye Yuan and Kris Kitani. 2020. Residual force control for agile human behavior imitation and extended motion synthesis. *arXiv preprint arXiv:2006.07364* (2020).