This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 899739

Technologies for computer-assisted crowd management

www.crowddna.eu

Call: H2020-FETOPEN-2018-2019-2020-01

Type of action: RIA

Grant agreement: 899739

WP N°3:	**Macro-to-micro motion analysis**
Deliverable Nº3.1:	**ML generator to analyze crowd movements in videos**
Task lead:	**URJC**
WP lead:	**URJC**
Version Nº:	**1.0**
Date:	**29/12/2023**

Document information	
Deliverable Nº and title:	ML generator to analyze crowd movements in videos
Version Nº:	0.3
Task lead:	URJC
WP lead:	URJC
Author(s):	Dan CASAS (URJC)
Reviewers:	He WANG (UCL/UL)
Submission date:	29/12/2023
Due date:	31/12/2023
Type:	DEM
Dissemination level:	PU

Document history

Date	Version	Author(s)	Comments
05/12/2023	0.1	Dan Casas	Creates and structures the document
18/12/2023	0.2	Dan Casas	First completed draft.
19/12/2023	0.2	He Wang	Review and comments.
29/12/2023	0.3	Dan Casas	Added conclusion, link to video.
29/12/2023	1.0	Solenne Fortun	Final layout.

Table of Contents

Table of Figures

Acronyms and Abbreviation

CDI	Crowd Dynamics International Limited
EC	European Commission
EMT	Executive Management team
FZJ	Forschungszentrum Julich Gmbh
GA	Grant Agreement
INRIA	Institut National De Recherche En Informatique Et Automatique
KPIs	Key Performance Indicators
ONH	Onhys
PO	Project Officer
UL	University of Leeds
ULM	Universität Ulm
URJC	Universidad Rey Juan Carlos
WP	Work-package

Executive Summary

This deliverable describes the efforts done during periods 1, 2 and half-way through 3 in the Work Package 3 (WP3) of the project towards, first, generating synthetic data and, second, training a detector. As a reminder, the overall objective of CrowdDNA WP3 is to deliver algorithms to detect micro-scale information of human interactions from macro scale observations such as crowd videos.

To this end, the consortium has worked on several key ingredients that leverage the crowd simulator described in Deliverable 2.1 to create synthetic data and annotations regarding human contact in dense crowds. In particular, we have extended our simulator to resolve the collisions that emerge when articulated 3D crowds are synthesized from 2D trajectories. Detecting and resolving collisions is a fundamental part of WP3, because it enables the computation of micro-scale inter-human interactions (e.g., pushes, contacts, etc.). Additionally, we have showed that collision-aware 3D crowds can also be obtained from 2D trajectories captured in the real world (*i.e.*, not simulated) and, more specifically, we have used data captured within the project in WP1 at FZJ facilities to train our models.

Data generated is used to train a machine learning model capable of inferring micro scale properties of a crowd observed from a surveillance camera. Importantly, we have tested our approach in real-world video footage, demonstrating that our solution generalizes well to unseen conditions.

1. 3D Crowd Generator

To model a dense 3D crowd, our approach can build on top of any existing 2D simulator that computes the global movement of a crowd by solving an optimization problem where each agent is approximated as a 2D particle, such as PLE [Guy et al. 2010] or RVO [Van den Berg *et al.* 2008]. Alternatively, we have also captured real-world 2D individual trajectories by tracking crowds under laboratory settings in the Work Package 1 (WP1), as we depict in **Figure** 1 (right), where orange hats ease the tracking of each individual. Animated 3D crowds are then generated by converting each 2D trajectory into a 3D articulated human encoded using the SMPL [Loper et al. 2015] body described later in Section 2.1, and depicted in **Figure** 1 (left). Special care needs to be taken to ensure that the resulting skeletal motion is plausible. In particular, we use cyclic motions to avoid producing visual artifacts repeatedly, which is especially crucial for locomotion to generate smooth walking cycles.

These 3D animations were handled using the Blend Tree functionality provided by the Unity Animation toolbox, which we use to convert 2D displacements into 3D pose sequences of locomotion or dancing sequences. Finally, we process the set of 3D pose sequences generated using the approach described in Section 2, enabling the synthesis of collision-aware realistic dense 3D crowds.

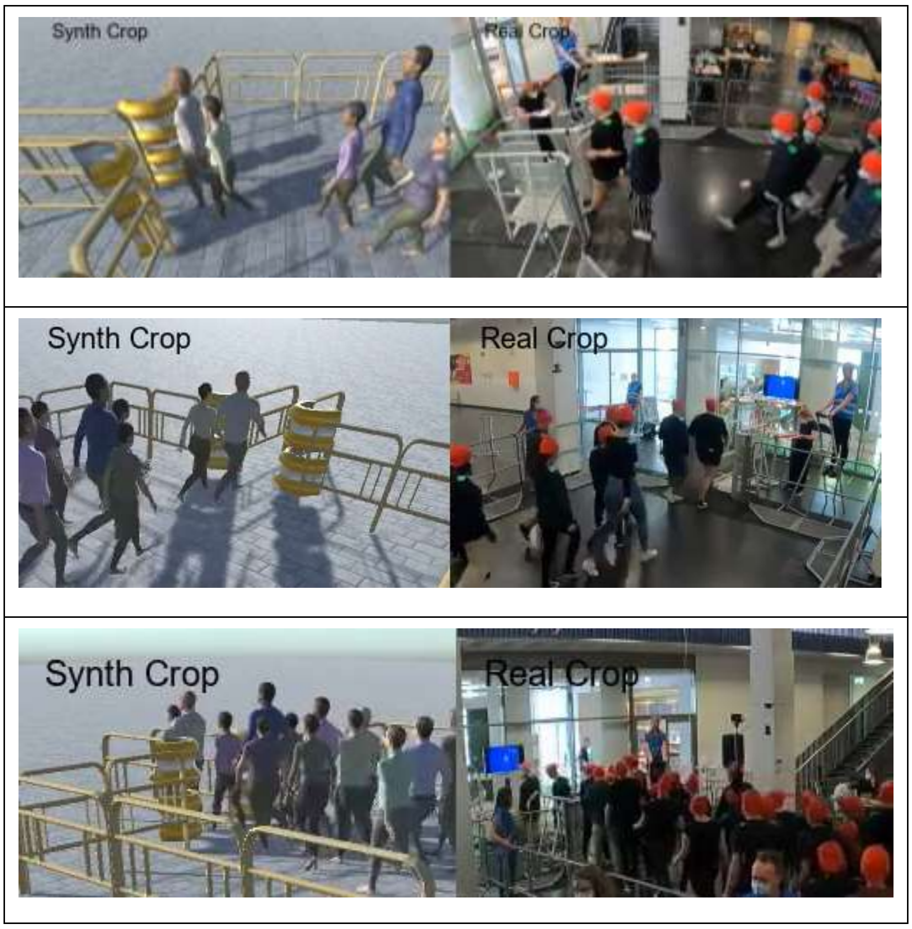

Figure 1. Real crowds captured within this project in a laboratory setting (right) are tracked to obtain 2D trajectories of each individual. We then convert 2D tracks into 3D articulated humans (left), which is used as training data for data-driven methods to analyze dense crowds.

2. Resolving Collisions in 3D Crowds

Our objective is to adjust the individual pose of many characters in a dense crowd by solving the collisions that exist between their bodies while maintaining the overall crowd motion as faithfully as possible. To this end, we first define our underlying parametric human model, which we then use to formulate a physics-based dynamic human model that is guided by motion capture data and the global crowd motion. Finally, we show how we incorporate collision handling into our formulation by adding a set of parametric volumetric primitives to approximate our human model.

2.1. Parametric Human Body Model

Starting from a human crowd consisting of a set of kinematic skeletons and their corresponding pose and global position over time, we first define a body representation that enables the computation of collisions between characters. To this end, we leverage the vast literature on 3D body models that deform a rigged parametric human template and, more specifically, use the nowadays standard SMPL [Loper *et al*. 2015] model.

Several previous works have extended SMPL's kinematic representation of the body surface to support physics-based soft-tissue deformation. However, at the target scale of our work, the relevance of skeletal response to collisions is far more notorious, and soft-tissue modeling does not scale well to the dozens of individuals required for dense crowds. Therefore, we favor the fast, responsive simulation of skeletal dynamics, and defer soft-tissue modeling to future work.

2.2. Physics-Based 3D Humans

To endow the above parametric human model with physics-based motion, we compute the pose θ as the result of a physics-based simulation. Specifically, we formulate equations of motion that include the following mechanical terms: inertia, gravity, joint constraints, collisions, and a control term to follow the input character body animation.

We formulate the motion of the articulated skeleton as a rigid body simulation with soft constraints to model joints. We denote as q the aggregate state of all individuals in a crowd, which includes the degrees of freedom of all rigid bones in each body. Given all mechanical terms for all bodies, we integrate the equations of motion using the popular optimization formulation of backward Euler. In our model, we use standard formulations for inertia, gravity and joints on rigid bones, and we discuss collisions and control below. We solve the optimization using Newton's method with analytical computation of gradients and Hessians.

2.3. Coupling to Full-Body Animations

The skeletal motion of individuals is dictated by pre-recorded full-body animations, and we want bodies to respond naturally to collisions while still following the input animation. Animation control is a whole research topic in its own, and we resort to a Proportional Derivative (PD) controller as it sufficed for our problem at hand.

Our PD controller couples each body to its corresponding skeletal animation input in the following way. For the body root, we couple absolute translation and rotation. For the rest of the body, on the other hand, we couple relative joint rotations. Given a joint with current body and joint rotations, and previous-step rotations, we define use a PD control that penalizes deviations in joint rotations and joint velocities between the physics-based simulation and the animation input. We found it was important to damp deviations in joint velocities, not absolute joint velocities, to ensure that bodies recover their trajectory smoothly after blocking collisions. For the pelvis and the feet, we use higher spring and damping coefficients to ensure stronger alignment of the trunk with the skeletal animation, as well as to avoid foot-skate.

2.4. Modeling Physics-Based Collision

Our physics-based 3D human simulation considers four types of collisions: body-ground, body-environment, inter-body, and intra-body collisions. An exact solution to collisions would require evaluating penetrations with respect to the exact surface model, but the deformable surface and skinning complicate this task. Instead, we propose to use a set of parametric geometric primitives to efficiently approximate the SMPL body volume and resolve all collision types.

To this end, for each bone of the SMPL kinematic skeleton we define a capsule computed as the intersection of two spheres and one cylinder. These capsule parameters are optimized such that the volume of the capsules maximize the overlap with the body volume. Capsules are transformed following the bone transformation given by pose parameter at each frame. Figure 2 presents a visualization of our choice of the parametric human model, depicting the original surface SMPL mesh (in solid brown) and our volumetric primitives on top (in semi-transparent). The use of a set of simple geometric proxies significantly simplifies the computation of collisions, while closely approximating the true volume of the character.

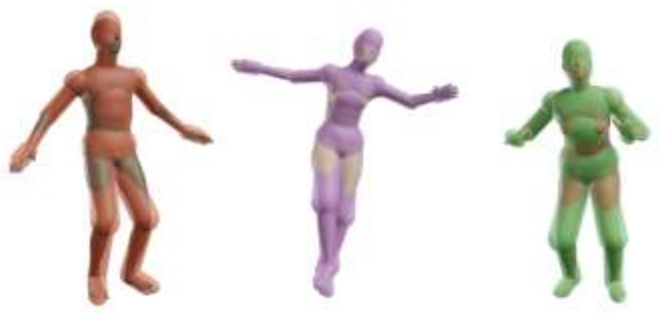

Figure 2. 3D human model (in brown, underneath) and the corresponding 3D proxies for 3 different body shapes and poses.

We execute collision detection for all four collision types as follows. For each body, we build an axis-aligned bounding boxe (AABB) after each simulation time step. For body-ground collision, we simply test all capsules against the ground plane and compute the penetration depth for colliding capsules. For body-environment collisions, we point-sample static environment objects, build a static AABB-tree, and cull collisions against body AABBs. For colliding points, we compute the penetration depth with respect to the collision capsule. For inter-body collisions, we first test body-level AABBs to cull body pairs. Finally, for intra-body collisions, we discard adjacent capsule pairs, which we identify as those that collide in T-pose. For each potentially colliding capsule-capsule pair (either inter-body or intra-body), we first execute a faster sphere-sphere culling test, and we compute capsule-capsule interpenetration only for those pairs that survive all culling tests.

Figure 3 illustrates the output of our collision-resolving step in a highly dense crowd simulating a concert environment. It can be seen that, despite the densely populated scenario, individuals do not intersect with each other, conveying a high-level of realism.

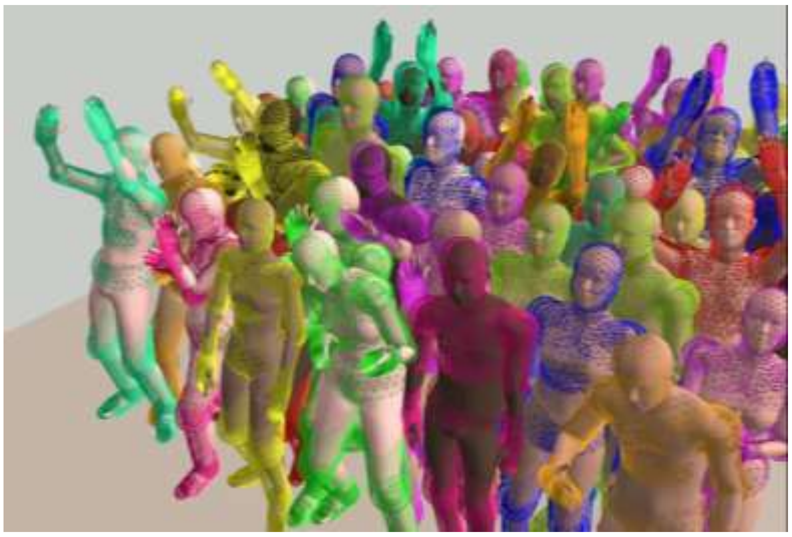

Figure 3. A 3D crowd simulating a concert scenario, processed with our physics-based collision avoidance. Our approach enables the generation of collision-free highly-dense dynamic crowds.

3. Micro-scale Crowd Analysis from Single Video

The 3D crowd generator described earlier in Sections 1 and 2 allows the synthesis of realistic 3D crowds that do not intersect with each other. Leveraging the capabilities of this simulator, we investigated the use of convolutional neural networks (CNN) to extract micro level information from videos crowds.

To this end, we first render the simulated 3D crowds using photorealistic 3D environments and human appearances, and extract optical flow features from the rendered videos. Optical flow computes the per-pixel intensity differences over time, which effectively encodes motion features of the persons that appear in the video. Our hypothesis is that micro-scale crowd information can be inferred from such optical flow features, but the main challenge was to ensure that the actual optical flow computation on synthetic frames (used for training) is, somehow, similar to real frames (used for testing). If this would not be the case, then training a network with synthetic data would never generalize to real data. Therefore, we evaluated the performance of optical flow in both synthetic and real images, and concluded that they are equivalent. Figure 4 depicts two frames (top), and they corresponding optical flow (bottom) for each data modality.

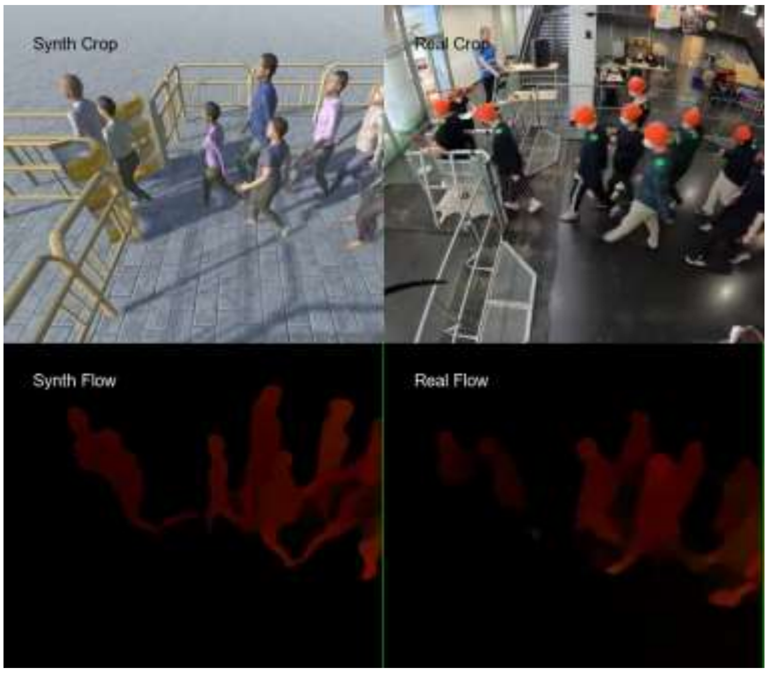

Figure 4. Estimated optical flow on synthetic (left) and real-world (right) data.

Using the optical flow as a input signal, our goal is to train an encoder-decoder convolutional neural network that outputs a per-pixel label that encodes some micro-scale information. Our model is general, because many types of information can be encoded as per-pixel labels: for example, contacts between individuals (i.e., label a given pixel if two persons are in contact), forecasting actions (i.e., label a given pixel with the probability of some type of action in the near future), etc. Figure 5 illustrates the architecture used for our macro-to-micro model.

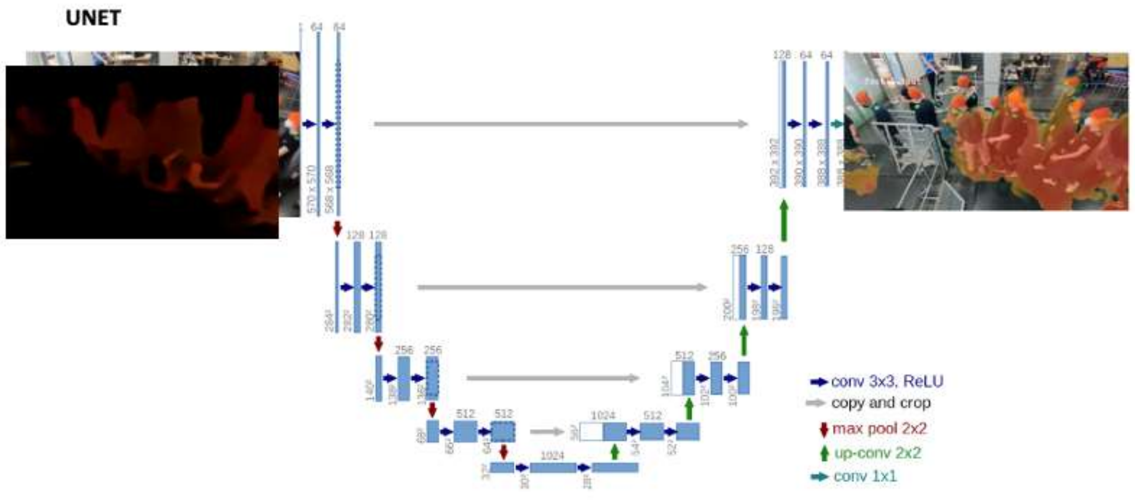

Figure 5. Convolutional Neural Network (CNN) encoder-decoder architecture. Our method takes as a input a optical flow image of the input frame (left) and outputs a per-pixel probability map (right)**.**

3.1. Individual Velocity Analysis

In our case, we have limited our first experiments to encoding per-pixel label that represents the probability that the person visible in such pixel is moving at the normal speed. In other words, we are want to infer a per-pixel color map that depicts if a person is moving as expected or not. To create the ground truth labels for such micro-scale event, we compute and store the 3D velocity of each agent in our simulator, and compute the normal distribution the velocity of all agents, for many scenarios and crowds. We then use this distribution to render colormaps that encode the probability of walking at abnormal velocities for the agent that appears in each pixel, for many videos from our simulation dataset. Our colormap representation have many advantages: it is treated as a normal RGB image, hence ideal for Convolutional Neutral Networks; it is human-friendly and straightforward to understand; and it can be used to encode many different features.

Figure 6 showcases two representative frames of our results in real-world data (i.e., these frame were not used to train our model). It can be seen that when the crowd is waiting to go through a bottleneck scenario (left), our network correctly identifies these situation as abnormal, hence it highlights in red the pixels were people are waiting. As the crowd passes through the bottleneck (right), pixels are not labeled as red, since people are moving. Likewise, pixels with no people waiting are also not highlighted in red.

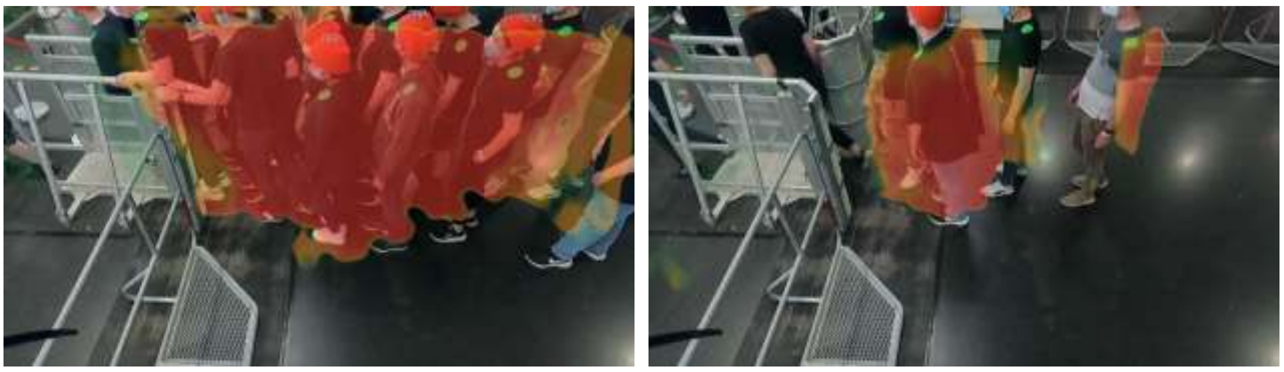

Figure 6. Qualitative results on real-world data. Our model outputs a colormap per-frame that highlights the parts of the image where persons are not moving at the normal velocity. Here, people waiting to go through the bottleneck scenario are detected.

3.2. Limb-level Collision Analysis

Since our micro-scale analysis model from Figure 5 is general, we can infer additional pixel-labels given an input image. A very interesting analysis, from a crowd management perspective, it is the inference of potential human-to-human collisions from videos. To this end, we have used our synthetic 3D crowd simulator to create animations of large crowds of different densities, ranging from rather comfortable situations with 2 persons square meter to highly-crowded situations. From such animations, we have leveraged our 3D volumetric proxies described in Section 2.4 to compute the 3D volume intersections of each individual of the crowd, and subsequently rendered these volumes from different camera views. This pipeline effectively creates limb-level contact maps encoded as per-pixel labels.

Figure 7 depicts human-to-human limb-level intersections in crowds as pixel labels from an external camera. We can then use these maps to train our UNet neural network and infer micro-level interactions from general videos.

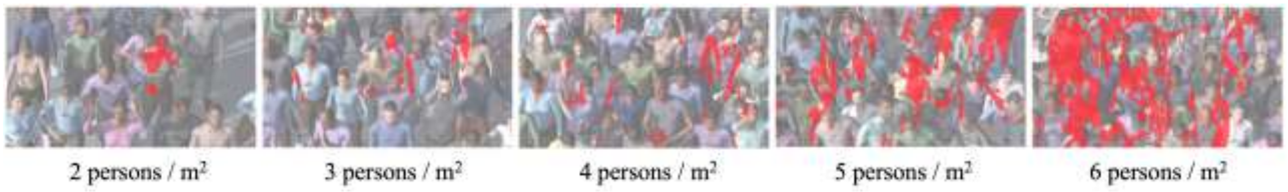

Figure 7. Volume intersection between individuals of crowds of different densities.

Conclusion

This deliverable has described the CrowdDNA project efforts towards generating and analyzing micro-level crowd events in general videos. Our pipeline covers the entire crowd simulation generation and synthesis, as well as the machine learning component to analyze different micro-level features from the crowd. We have demonstrated that our pipeline effectively estimates micro-level information both in synthetic and real-world images. Please see the additional video[1] here for animated results.

[1] *https://youtu.be/dCo3yqJtsWU*

References

[1] Guy, Stephen J., Jatin Chhugani, Sean Curtis, Pradeep Dubey, Ming C. Lin, and Dinesh Manocha. "PLEdestrians: A Least-Effort Approach to Crowd Simulation." In *Symposium on Computer Animation (SCA)*, pp. 119-128. 2010.

[2] Van den Berg, Jur, Ming Lin, and Dinesh Manocha. "Reciprocal velocity obstacles for real-time multi-agent navigation." In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1928-1935., 2008.

[3] Loper, M., Naureen, M., Romero, J., Pons-Moll, G., and Black, Michael J. "SMPL: A Skinned Multi-Person Linear Model". In *ACM Transactions on Graphics (TOG)*, 2015.