Photo Wake-Up: 3D Character Animation from a Single Photo

Chung-Yi Weng, Brian Curless, Ira Kemelmacher-Shlizerman

UW Reality Lab
Paul G. Allen School of Computer Science and Engineering
University of Washington

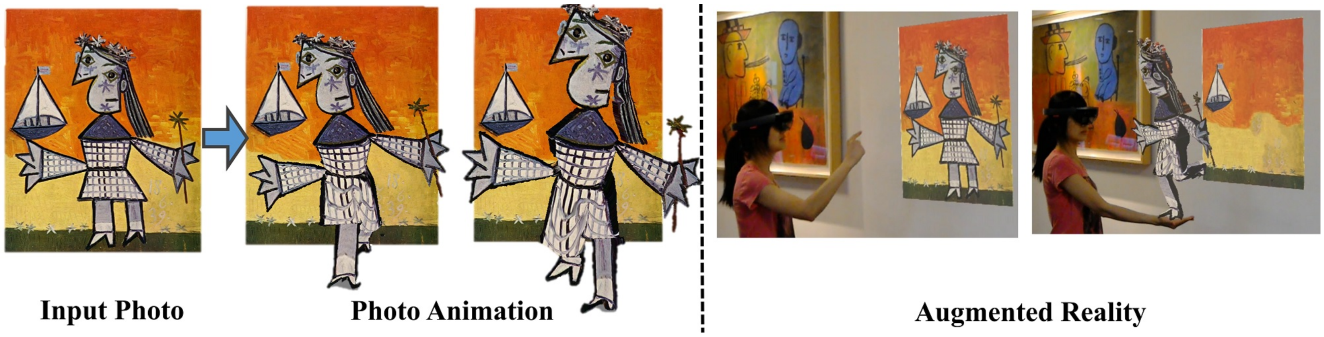


Figure 1: Given a single photo as input (far left), we create a 3D animatable version of the subject, which can now walk towards the viewer (middle). The 3D result can be experienced in augmented reality (right); in the result above the user has virtually hung the artwork with a HoloLens headset and can watch the character run out of the painting from different views. **Please see all results in the supplementary video:** `https://youtu.be/G63goXc5MyU`. *Photo credit: wikiart.org*.

Abstract

We present a method and application for animating a human subject from a single photo. E.g., the character can walk out, run, sit, or jump in 3D. The key contributions of this paper are: 1) an application of viewing and animating humans in single photos in 3D, 2) a novel 2D warping method to deform a posable template body model to fit the person's complex silhouette to create an animatable mesh, and 3) a method for handling partial self occlusions. We compare to state-of-the-art related methods and evaluate results with human studies. Further, we present an interactive interface that allows re-posing the person in 3D, and an augmented reality setup where the animated 3D person can emerge from the photo into the real world. We demonstrate the method on photos, posters, and art. The project page is at `https://grail.cs.washington.edu/projects/wakeup/`.

1. Introduction

> Whether you come back by page or by the big screen,
> Hogwarts will always be there to welcome you home.
>
> *J.K. Rowling*

In this paper, we propose to "wake up a photo" by bringing the foreground character to life, so that it can be animated in 3D and emerge from the photo. Related to our application are cinemagraphs and GIFs[1] where a small motion is introduced to a photo to visualize dominant dynamic areas. Unlike a cinemagraph, which is a 2D experience created from video, our method takes a single photo as input and results in a fully 3D experience. The output animation can be played as a video, viewed interactively on a monitor, and as an augmented or virtual reality experience, where a user with an headset can enjoy the central figure of a photo coming out into the real world.

A central challenge in delivering a compelling experience is to have the reconstructed subject closely match the silhouette of the clothed person in the photo, including self-occlusion of, e.g., the subject's arm against the torso. Our approach begins with existing methods for segmenting a person from an image, 2D skeleton estimation, and fitting a (semi-nude) morphable, posable 3D model. The result of this first stage, while animatable, does not conform to the silhouette and does not look natural.

Our key technical contribution, then, is a method for constructing an animatable 3D model that matches the silhouette in a single photo and handles self-occlusion. Rather

[1] Artistic cinemagraphs: `http://cinemagraphs.com/`

than deforming the 3D mesh from the first stage – a difficult problem for intricate regions such as fingers and for scenarios like abstract artwork – we map the problem to 2D, perform a silhouette-aligning warp in image space, and then lift the result back into 3D. This 2D warping approach works well for handling complex silhouettes. Further, by introducing label maps that delineate the boundaries between body parts, we extend our method to handle certain self-occlusions.

Our operating range on input and output is as follows. The person should be shown in whole (full body photo) as a fairly frontal view. We support partial occlusion, specifically of arms in front of the body. While we aim for a mesh that is sufficient for convincing animation, we do not guarantee a metrically correct 3D mesh, due to the inherent ambiguity in reconstructing a 3D model from 2D input. Finally, as existing methods for automatic detection, segmentation, and skeleton fitting are not yet fully reliable (esp. for abstract artwork), and hallucinating the appearance of the back of a person is an open research problem, we provide a user interface so that a small amount of input can correct errors and guide texturing when needed or desired.

To the best of our knowledge, our system is the first to enable 3D animation of a clothed subject from a single image. The closest related work either does not recover fully 3D models [26] or is built on monocular video input [11]. We compare to these prior approaches, and finally show results for a wide variety of examples as 3D animations and AR experiences.

2. Related Work

General animation from video has led to many creative effects over the years. The seminal "Video Textures" [46] work shows how to create a video of infinite length starting from a single video. Human-specific video textures were produced from motion capture videos via motion graphs [22]. [51] explore multi-view captures for human motion animation, and [54] demonstrate that clothing can be deformed in user videos guided by body skeleton and videos of models wearing the same clothing. Cinemagraphs [47, 13] or Cliplets [28] create a still with small motion in some part of the still, by segmenting part of a given video in time and space.

Relevant also are animations created from big data sets of images, e.g., personal photo collections of a person where the animation shows a transformation of a face through years [32], or Internet photos to animate transformation of a location in the world through years [40], e.g., how flowers grow on Lombard street in San Francisco, or the change of glaciers over a decade.

Animating from a single photo, rather than videos or photo collections, also resulted in fascinating effects. [18] animate segmented regions to create an effect of water ripples or swaying flowers. [52] predict motion cycles of animals from a still photo of a group of animals, e.g., a group of birds where each bird has a different wing pose. [33] show that it's possible to modify the 3D viewpoint of an object in a still by matching to a database of 3D shapes, e.g., rotating a car on in a street photo. [12] showed how to use a video of an actor making facial expressions and moving their head to create a similar motion in a still photo. Specific to body shapes, [53] showed that it's possible to change the body weight and height from a single image and in a full video [27]. [26] presented a user-intensive, as-rigid-as-possible 2D animation of a human character in a photo, while ours is 3D.

For 3D body shape estimation from single photo, [16] provided the SMPL model which captures diverse body shapes and proved highly useful for 3D pose and shape estimation applications. Further, using deep networks and the SMPL model, [49, 29, 43, 42] present end-to-end frameworks for single view body pose and shape estimation. [48] directly infer a volumetric body shape. [24] finds dense correspondence between human subjects and UV texture maps. For multi-view, [37, 20] reconstruct a 3D mesh from sketches or silhouettes. [11] applied SMPL model fitting to a video of a subject rotating in front of a static camera, and is further extended in [10] to improve mesh and texture quality with shape from shading. Recently, the idea of parametric model has further been extended from humans to animals [55, 30].

Most single-image person animation has focused on primarily 2D or pseudo-3D animation (e.g., [26]) while we aim to provide a fully 3D experience. Most methods for 3D body shape estimation focus on semi-nude body reconstruction and not necessarily ready for animation, while we take cloth into account and look for an animatable solution. The most similar 3D reconstruction work is [11] although they take a video as input. We compare our results to [26] and [11] in Sec. 6.

3. Overview

Given a single photo, we propose to animate the human subject in the photo. The overall system works as follows (Fig. 2): We first apply state-of-the-art algorithms to perform person detection, segmentation, and 2D pose estimation. From the results, we devise a method to construct a rigged mesh (Section 4). Any 3D motion sequence can then be used to animate the rigged mesh.

To be more specific, we use Mask R-CNN [25] for person detection and segmentation (implementation by [41]). 2D body pose is estimated using [50], and person segmentation is refined using Dense CRF [35]. Once the person is segmented out of the photo, we apply PatchMatch [14] to fill in the regions where the person used to be.

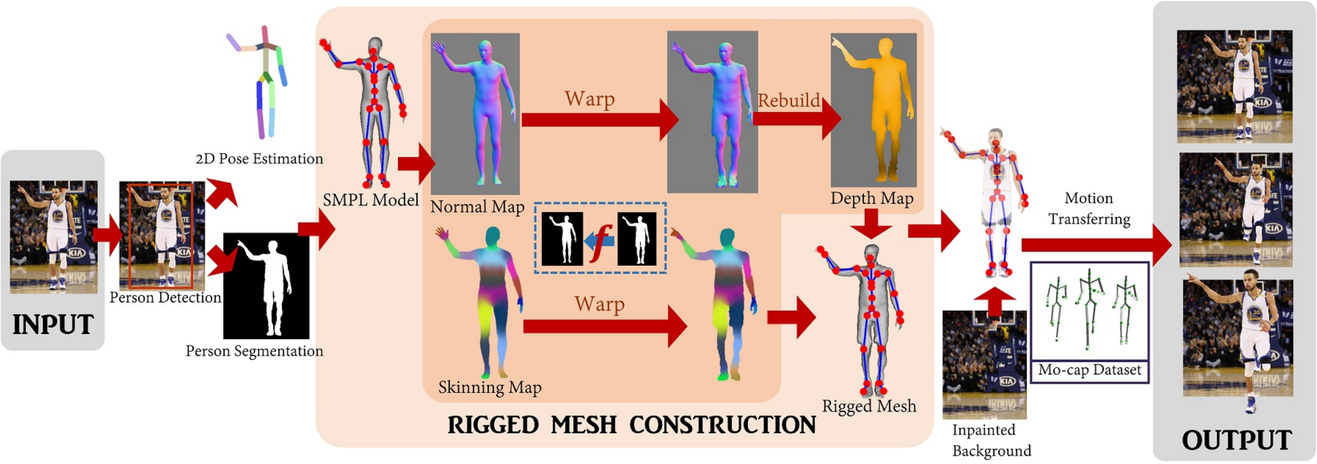


Figure 2: Overview of our method. Given a photo, person detection, 2D pose estimation, and person segmentation, is performed using off-the-shelf algorithms. Then, A SMPL template model is fit to the 2D pose and projected into the image as a normal map and a skinning map. The core of our system is: find a mapping between person's silhouette and the SMPL silhouette, warp the SMPL normal/skinning maps to the output, and build a depth map by integrating the warped normal map. This process is repeated to simulate the model's back view and combine depth and skinning maps to create a complete, rigged 3D mesh. The mesh is further textured, and animated using motion capture sequences on an inpainted background. *Photo credit: gettyimages*

4. Mesh Construction and Rigging

The key technical idea of this paper is how to recover an animatable, textured 3D mesh from a single photo to fit the proposed application.

We begin by fitting the SMPL morphable body model [36] to a photo, including the follow-on method for fitting a shape in 3D to the 2D skeleton [16]. The recovered SMPL model provides an excellent starting point, but it is semi-nude, does not conform to the underlying body shape of the person and, importantly, does not match the clothed silhouette of the person.

One way is to force the SMPL model to fit the silhouettes by optimizing vertex locations on the SMPL mesh, taking care to respect silhouette boundaries, avoid pinching, and self-intersection. This is challenging especially around intricate regions such as fingers. This was indeed explored by [11], and we compare to those results in the experiments.

Instead, we take a 2D approach: warp the SMPL silhouette to match the person silhouette in the original image and then apply that warp to projected SMPL normal maps and skinning maps. The resulting normal and skinning maps can be constructed for both front and (imputed) back views and then lifted into 3D, along with the fitted 3D skeleton, to recover a rigged body mesh that exactly agrees with the silhouettes, ready for animation. The center box in Figure 2 illustrates our approach.

In the following, we describe how we construct a rigged mesh using 2D warping (Section 4.1), then present how to handle arm-over-body self-occlusion (Section 4.2).

4.1. Mesh Warping, Rigging, & Skinning

In this section, we describe the process for constructing a rigged mesh for a subject without self-occlusion.

We start with the 2D pose of the person and the person's silhouette mask S. For simplicity, we refer to S both as a set and as a function, i.e., as the set of all pixels within the silhouette, and as a binary function $S(x) = 1$ for pixel x inside the silhouette or $S(x) = 0$ for x outside the silhouette.

To construct a 3D mesh with skeletal rigging, we first fit a SMPL model to the 2D input pose using the method proposed by [16], which additionally recovers camera parameters. We then project this mesh into the camera view to form a silhouette mask S_{SMPL}. The projection additionally gives us a depth map $Z_{\text{SMPL}}(x)$, a normal map $N_{\text{SMPL}}(x)$ and a skinning map $W_{\text{SMPL}}(x)$ for pixels $x \in S_{\text{SMPL}}$. The skinning map is derived from the per-vertex skinning weights in the SMPL model and is thus vector-valued at each pixel (one skinning weight per bone).

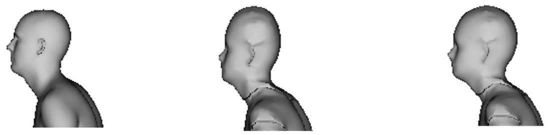
Guided by S_{SMPL} and the input photo's silhouette mask S, we then warp Z_{SMPL}, N_{SMPL}, and W_{SMPL} to construct an output depth map (at the silhouette only) $Z_{\partial S}(x \in \partial S)$, normal map $N(x)$, and skinning map $W(x)$, respectively, for pixels $x \in S$. $N(x)$ is then integrated to recover the final depth map $Z(x)$, subject to matching $Z_{\partial S}(x)$ at the silhouette boundary ∂S. More concretely, we solve for a smooth inverse warp, $f(x)$, such that:

$$S(x) = S_{\text{SMPL}}(f(x)) \tag{1}$$

and then apply this warp to the depth and skinning maps:

$$\begin{aligned} Z_{\partial S}(x \in \partial S) &= Z_{\text{SMPL}}(f(x)) \qquad (2)\\ N(x) &= N_{\text{SMPL}}(f(x)) \qquad (3)\\ Z(x) &= \text{Integrate}[N; Z_{\partial S}] \qquad (4)\\ W(x) &= W_{\text{SMPL}}(f(x)) \qquad (5)\end{aligned}$$

We experimented with setting $Z(x) = Z_{\text{SMPL}}(f(x))$, but the resulting meshes were usually too flat in the z direction (See Fig. 3b). The warping procedure typically stretches the geometry in the plane (the SMPL model is usually thinner than the clothed subject, often thinner than even the unclothed subject), without similarly stretching (typically inflating) the depth. We address this problem by instead warping the *normals* to arrive at $N(x)$ and then integrating them to produce $Z(x)$. In particular, following [15], we solve a sparse linear system to produce a $Z(x)$ that agrees closely with the warped normals $N(x)$ subject to the boundary constraint that $Z(x) = Z_{\partial S}(x)$ for pixels $x \in \partial S$. Fig. 3 shows the difference between the two methods we experimented with.



(a) SMPL (b) Warp depth (c) Warp normal and then integrate

Figure 3: Comparison of different depth map constructions, after stitching front and back depth maps together (Section 4.1.3). Given (a) a reference SMPL model, we can reconstruct a mesh (b) by warping the SMPL depth maps or (c) by warping the SMPL normal maps and then integrating. Notice the flattening evident in (b), particularly around the head.

To construct the inverse warp, $f(x)$, many smooth warping functions are possible; we choose one based on mean-value coordinates [23] because it is well defined over the entire plane for arbitrary planar polygons without self-intersections, which fits our cases very well. In particular, given the ordered set of points (vertices) on the closed polygonal boundary of the input silhouette, $p_i \in \partial S = (p_0, p_1, \ldots, p_{m-1})$, we can represent any point inside of S as:

$$x = \sum_{i=0}^{m-1} \lambda_i(x) p_i \tag{6}$$

where $(\lambda_0(x), \lambda_1(x), \ldots, \lambda_{m-1}(x))$ are the mean-value coordinates of any $x \in S$ with respect to the boundary vertices p_i.

Suppose we have a correspondence function ϕ that identifies p_i on the input silhouette boundary ∂S with points on the SMPL silhouette boundary $p_i^{\text{SMPL}} \in \partial S_{\text{SMPL}} = (p_0^{\text{SMPL}}, p_1^{\text{SMPL}}, \ldots, p_{n-1}^{\text{SMPL}})$:

$$p_i \rightarrow p_{\phi[i]}^{\text{SMPL}}. \tag{7}$$

Then, using the same mean-value coordinates from Eq. 6, we define the warp function to be:

$$f(x) = \sum_{i=0}^{m-1} \lambda_i(x) p_{\phi[i]}^{\text{SMPL}}. \tag{8}$$

Next, we describe how we compute the correspondence function ϕ, fill holes in the normal and skinning maps, and then construct a complete mesh with texture.

4.1.1 Boundary matching

We now seek a mapping ϕ that provides correspondence between points $p_i \in \partial S$ and points $p_j^{\text{SMPL}} \in \partial S_{\text{SMPL}}$. We would like each point p_i to be close to its corresponding point $p_{\phi[i]}^{\text{SMPL}}$, and, to encourage smoothness, we would like the mapping to be monotonic without large jumps in the indexing. To this end, we solve for $\phi[i]$ to satisfy:

$$\underset{\phi[0],\ldots,\phi[m-1]}{\arg\min} \sum_{i=0}^{m-1} D(p_i, p_{\phi[i]}^{\text{SMPL}}) + T(\phi[i], \phi([i+1])) \tag{9}$$

where

$$D(p_i, p_{\phi[i]}^{\text{SMPL}}) = \| p_i - p_{\phi[i]}^{\text{SMPL}} \|_2 \tag{10}$$

and

$$T(\phi[i], \phi[i+1]) = \begin{cases} 1, & \text{if } 0 \leq \phi[i+1] - \phi[i] \leq \kappa \\ \infty, & \text{otherwise} \end{cases} \tag{11}$$

$D(p_i, p_{\phi[i]}^{\text{SMPL}})$ is designed to encourage closeness of corresponding points, and $T(\phi[i], \phi[i+1])$ avoids generating an out-of-order sequence with big jumps. Because we are indexing over closed polygons, we actually use $\phi[i\%m]\%n$ in the objective. With $\kappa = 32$, we solve for ϕ with dynamic programming.

4.1.2 Hole-filling

In practice, holes may arise when warping by $f(x)$, i.e., small regions in which $f(x) \notin S_{\text{SMPL}}$, due to non-bijective mapping between ∂S and ∂S_{SMPL}. We smoothly fill these holes in the warped normal and skinning weight maps. Please refer to the supplemental material for more detail and illustration of the results of this step.

4.1.3 Constructing the complete mesh

The method described so far recovers depth and skinning maps for the front of a person. To recover the back of the person, we virtually render back view of the fitted SMPL model, mirror the person mask, and then apply the warping method described previously.

We reconstruct front and back meshes in the standard way: back-project depths into 3D and construct two triangles for each 2x2 neighborhood. We assign corresponding skinning weights to each vertex. Stitching the front and back meshes together is straightforward as they correspond at the boundary. Fig. 4 illustrates the front and back meshes and the stitched model.

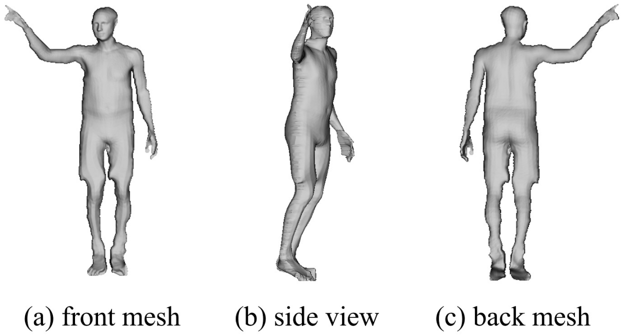

(a) front mesh (b) side view (c) back mesh

Figure 4: Reconstructed mesh results. We reconstruct the front mesh (a) and the back mesh (c) separately and then combine them into one mesh, viewed from the side in (b).

4.2. Self-occlusion

When the subject self-occludes – one body part over another – reconstructing a single depth map (e.g., for the front) from a binary silhouette will not be sufficient. To handle self-occlusion, we segment the body into parts via body label map, complete the partially occluded segments, and then reconstruct each part using the method described in Section 4.1. Fig. 5 illustrates our approach.

We focus on self-occlusion when the arms partially cross other body parts such that the covered parts are each still a single connected component. Our method does not handle all self-occlusion scenarios, but does significantly extend the operating range and show a path toward handling more cases.

4.2.1 Body label map

The projected SMPL model provides a reference body label map L_{SMPL} that does not conform closely to the image. We use this label map to construct a final label map L in two stages: (1) estimate an initial label map L_{init} for each pixel $x \in S$ to be as similar as possible to L_{SMPL}, then (2) refine L_{init} at occlusion boundaries where the label discontinuities should coincide with edges in the input image.

Initial Body Labeling. We solve for the initial (rough) body label map L_{init} by minimizing a Markov Random Field (MRF) objective:

$$\min_{L_{\text{init}}} \sum_{p\in S} U(L_{\text{init}}(p)) + \gamma \sum_{p\in S, q\in\mathcal{N}(p)\cap S} V(L_{\text{init}}(p), L_{\text{init}}(q)) \quad (12)$$

where

$$U(L_{\text{init}}(p)) = \min_{r|L_{\text{SMPL}}(r)=L(p)} \|p-r\|_2 \quad (13)$$

$$V(L_{\text{init}}(p), L_{\text{init}}(q)) = \begin{cases} 1 & \text{if } L_{\text{init}}(p) \neq L_{\text{init}}(q) \\ 0 & \text{otherwise} \end{cases} \quad (14)$$

$\mathcal{N}(p)$ is the 8-neighborhood of p. $U(.)$ scores a label according to the distance to the nearest point in L_{SMPL} with the same label, thus encouraging L_{init} to be similar in shape to L_{SMPL}, while $V(.)$ encourages spatially coherent labels.

We use α-expansion [17] to approximately solve for L_{init}, with $\gamma = 16$. Fig. 5(b) illustrates the initial label map produced by this step.

Refined Body Labeling. Next, we refine the body label map to more cleanly separate occlusion boundaries.

Occlusion boundaries occur when two pixels with different part labels are neighbors in the image, but are not neighbors on the 3D body surface. To identify these pixels, we first compute warp functions f_ℓ that map each body part $L_{\text{init}} = \ell$ to the corresponding body part $L_{\text{SMPL}} = \ell$, using the mean-value coordinate approach described in Section 4.1, performed part-by-part. Then, along the boundaries of arm parts of L_{init}, for each pair of neighboring pixels (p, q) with different labels, we determine the corresponding projected SMPL locations $(f_{L_{\text{init}}(p)}(p), f_{L_{\text{init}}(q)}(q))$, back-project them onto the SMPL mesh, and check if they are near each other on the surface. If not, these pixels are identified as occlusion pixels. Finally, we dilate around these occlusion pixels to generate an occlusion mask O. The result is shown in Fig. 5(c).

We now refine the labels within O to better follow color discontinuities in the image I, giving us the final body label map L. For this, we define another MRF:

$$\min_{L} \sum_{p\in O} U(L(p)) + \gamma \sum_{p\in O, q\in\mathcal{N}(p)} V(L(p), L(q)) \quad (15)$$

where

$$U(L(p)) = -\log(\text{GMM}(L(p), I(p))) \quad (16)$$

$$V(L(p), L(q)) = C(L(p), L(q))e^{-\beta\|I(p)-I(q)\|^2} \quad (17)$$

$$C(L(p), L(q)) = \begin{cases} 1/\|p-q\| & \text{if } L(p) \neq L(q) \\ 0 & \text{otherwise} \end{cases} \quad (18)$$

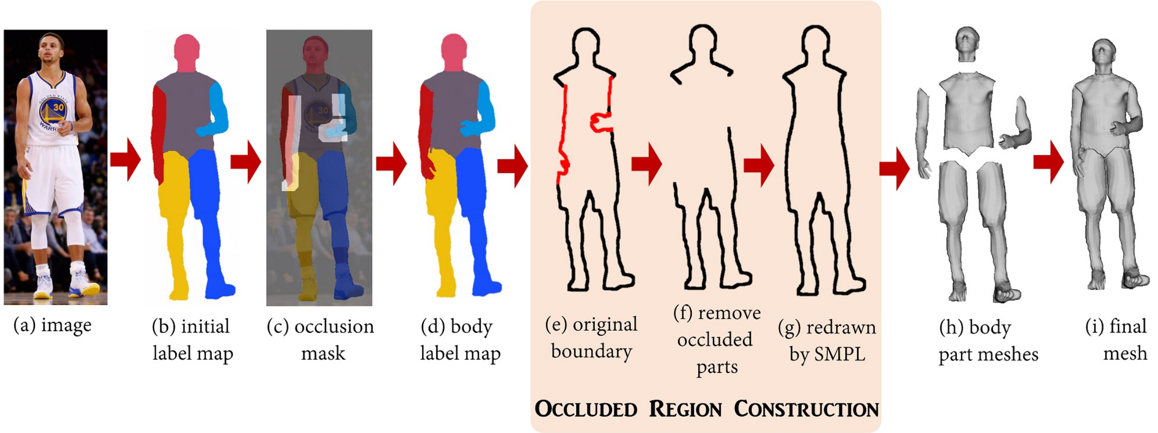


Figure 5: Starting from the input image (a) and its corresponding silhouette and projected SMPL body part model, we recover an initial body part label map (b). After identifying points at occlusion boundaries, we construction an occlusion mask (lighter areas in (c)) and then refine it to construct the final body label map (d). The body part regions near occlusions have spurious boundaries, shown in red in (e). We remove these spurious boundaries (f) and replace them with transformed versions of the SMPL boundaries (g). We then rebuild the body part-by-part (h) and assemble into the final mesh (i). *Photo credit: gettyimages*

where $\mathrm{GMM}(L(p), I(p))$ is the probability of p with color $I(p)$ labeled as $L(p)$, modeled using a Gaussian Mixture Model. We set $\gamma = 8$, and, following [45], we set β to be:

$$\beta = (2\langle\|(I(p) - I(q)\|^2\rangle)^{-1} \tag{19}$$

where $\langle.\rangle$ averages over all pairs of neighboring pixels in O.

The problem is solved by iteratively applying α-expansions [17], where in each iteration we re-estimate $\mathrm{GMM}(.)$ using the latest approximated L initizlied as L_{init}. Fig 5(d) illustrates the final body map.

4.2.2 Occluded region construction

We now have the shapes of the unoccluded segments; the next challenge is to recover the shapes of the partially occluded parts.

We first combine the labels of the head, torso, and legs together into one region B. Then we extract the boundary ∂B and identify the occlusion boundaries, $\partial B^{ocl} \in \partial B \cap O$ (shown in red in Fig. 5(e)). Next, for a contiguous set of points $\partial B_i^{ocl} \in \partial B^{ocl}$ (e.g., one of the three separate red curves in Fig. 5(e)), we find the corresponding boundary $\partial B_{\mathrm{SMPL}}^{ocl} \in \partial B_{\mathrm{SMPL}}$ using the the boundary matching algorithm from Section 4.1.1, where B_{SMPL} is the region formed by projecting the SMPL head, torso, and legs to the image plane. We then replace ∂B_i^{ocl} with $\partial B_{\mathrm{SMPL}}^{ocl}$ by a similarity transform defined by the end points of ∂B_i^{ocl} and $\partial B_{\mathrm{SMPL}}^{ocl}$, as shown in Fig.5-(f) and (g).

4.2.3 Mesh construction

Once we have completed body labeling and recovered occluded shapes, we project the SMPL model part-by-part to get per-part SMPL depth, normal, and skinning weight maps, then follow the approach in Section 4.1 to build part meshes (Fig.5-(h)), and assemble them together to get our final body mesh (Fig.5-(i)). Finally, we apply Laplacian smoothing to reduce jagged artifacts along the mesh boundaries due to binary silhouette segmentation.

5. Final Steps

Head pose correction: Accuracy in head pose is important for good animation, while the SMPL head pose is often incorrect. Thus, as in [34, 31], we detect facial fiducials in the image and solve for the 3D head pose that best aligns the corresponding, projected 3D fiducials with the detected ones. After reconstructing the depth map for the head as before, we apply a smooth warp that exactly aligns the projected 3D fiducials to the image fiducials. Whenever the face or fiducials are not detected, this step is skipped.

Texturing: For the front of the subject, we project the image onto the geometry. Occluded, frontal body part regions are filled using PatchMatch [14]. Hallucinating the back texture is an open research problem [38, 21, 39]. We provide two options: (1) paste a mirrored copy of the front texture onto the back, (2) inpaint with optional user guidance. For the second option, inpainting of the back is guided by the body label maps, drawing texture from regions with the same body labels. The user can easily alter these label maps to, e.g., encourage filling in the back of the head with hair texture rather than face texture. Finally the front and back textures are stitched with poisson blending [44].

Please refer to the supplemental material for more details and illustrations of head pose correction and texturing.

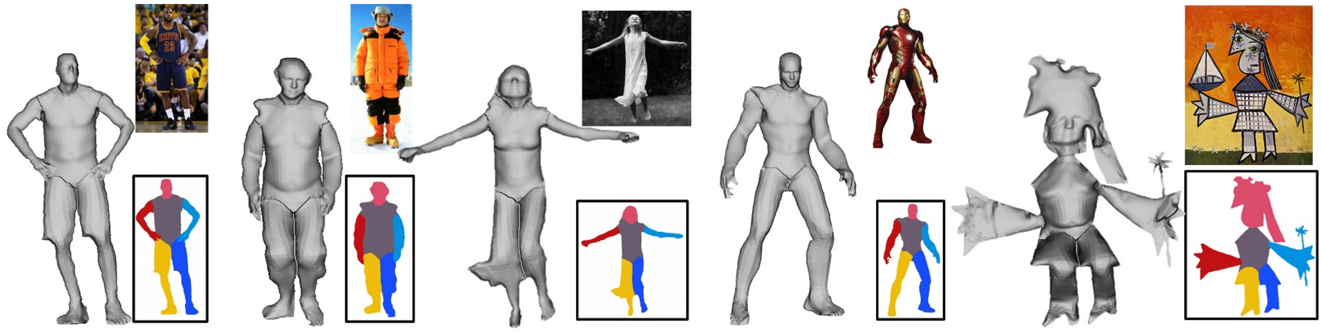


Figure 6: Examples of body label maps and meshes (input photos are put on top right corner). *Photo credits: [3, 5, 4, 6, 8]*

6. Results and Discussion

Below we describe our user interface, results, and comparisons to related methods. We have tested our method on 70 photos downloaded from the Internet (spanning art, posters, and graffiti that satisfied our photo specifications–full body, mostly frontal). Figs. 7 and 8 show our typical animation and augmented reality results. With our UI, the user can change the viewpoint during animation, and edit the human pose. With an AR headset, the user can place the artwork on the wall and walk around the animation while it is playing. **Please refer to the supplementary video[2] for dynamic versions of the results.**

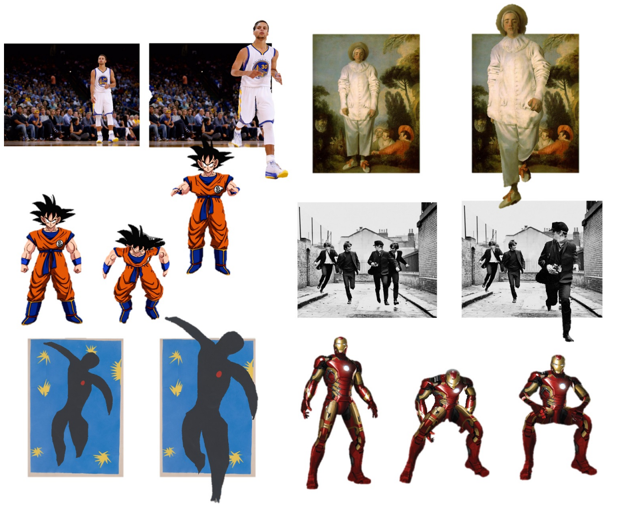


Figure 7: Six animation results. The input is always on left. *Photo credits: [3, 9, 2, 1, 8, 6]*

User interface: We have created a user interface where the user can interactively: (1) Modify the animation: the default animation is "running", where the user can keep some body parts fixed, change the sequence (e.g., choose any sequence from [19]), modify pose and have the model perform an action starting from the modified pose. (2) Improve the automatic detection box, skeleton, segmentation, and body label map if they wish. (3) Choose to use mirrored textures for the back or make adjustments via editing of the body label map. The user interaction time for (2) and (3) is seconds, when needed.

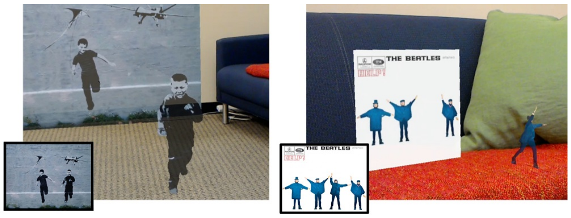
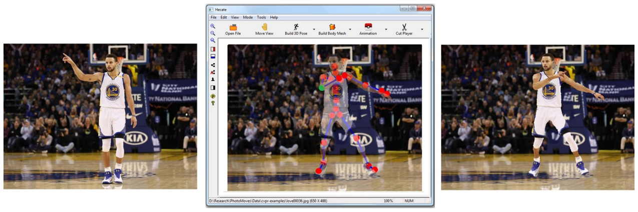


Figure 8: AR results of our method for different environments (input photos inset). The floor (left) and couch (right) are real, while the people are augmented. *Photo credits: [7, 1]*

Fig. 9 shows an example of the pose editing process. In our UI the mesh becomes transparent to reveal the body skeleton. By selecting and dragging the joints the user can change the orientation of the corresponding bones. A new image where the pose is edited can be then easily generated.



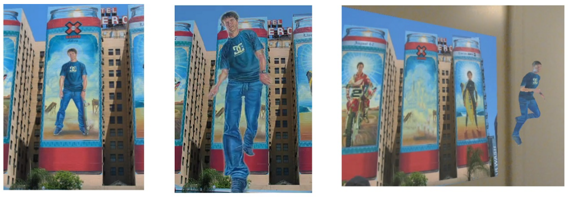
(a) image (b) UI to drag joints (c) results

Figure 9: Our pose editing UI: (a) Input photo. (b) Editing pose by dragging joints. (c) Result. *Photo credit: [3]*

The underlying reconstructed geometry for several examples is shown in Fig. 6. The resulting meshes do not necessarily represent the exact 3D geometry of the underlying subject, but they are sufficient for animation in this application and outperform state of the art as shown below.

Comparison with [26]: We have run our method on

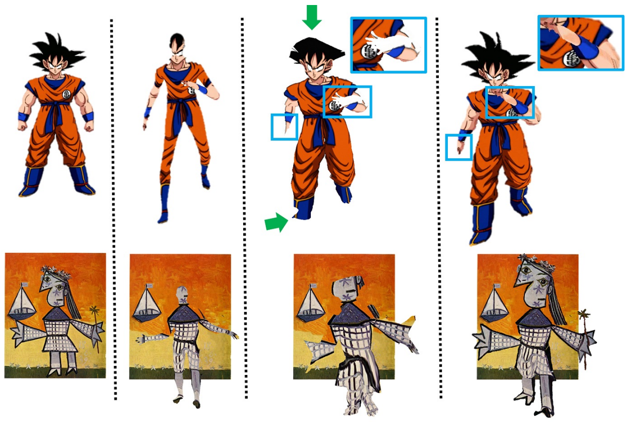
[2] https://youtu.be/G63goXc5MyU



(a) input photo (b) Hornung et al. (c) ours – 3D demonstration

Figure 10: Comparison result with [26]: (a) input photo; (b) animation method proposed in [26]; (c) 3D demonstration using our method, which is not possible in [26]. *Photo credit: Hornung et al.*

the only example in [26] that demonstrated substantial out-of-plane motion rather than primarily in-plane 2D motion (see Fig. 10). Our result appears much less distorted in still frames (due to actual 3D modeling) and enables 3D experiences (e.g., AR) not possible in [26]. We verified our qualitative observation with a user study on MTurk, asking users to decide which animation is "more realistic." 103 participants responded, and 86% preferred ours.

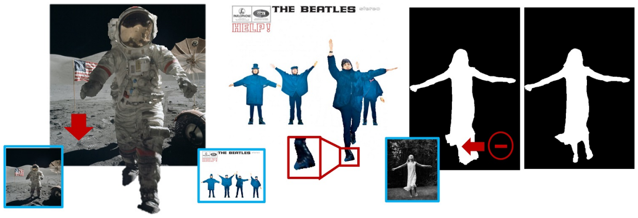


(a) input (b) Bogo et al. (c) Alldieck et al. (d) our method

Figure 11: Comparison with [16, 11]. (a) Input photo. (b) A fitted SMPL model [16]. (c) A deformed mesh using [11]. The mesh fails to deform hair and shoes (green arrows) and fingers (blue box). (d) Our mesh. *Photo credits: [2, 8]*

Comparison with Bogo et al. [16]: As shown in Fig. 11(b), the fitted, semi-nude SMPL model [16] does not correctly handle subject silhouettes.

Comparison with Alldieck et al. [11]: In [11], a SMPL mesh is optimized to approximately match silhouettes of a rotating human subject in a monocular video sequence. Their posted code uses 120 input frames, with objective weights tuned accordingly; we thus provide their method with 120 copies of the input image, in addition to the same 2D person pose and segmentation we use. The results are shown in Fig. 11(c). Their method does not fit the silhouette well; e.g., smooth SMPL parts don't become complex (bald head mapped to big, jagged hair) and the detailed fingers are not warped well to the closed fists or abstract art hands. These failures are mostly due to a strong human shape prior, which is not suited to tackling cases like cartoon characters or abstract paintings.



(a) (b) (c)

Figure 12: Examples of limitations (inputs in blue boxes). (a) Shadows not modeled. (b) Unrealistic mesh. (c) manually correcting segmentation errors. *Photo credits: [9, 1, 4]*

Limitations: We note the following limitations (see also Fig. 12): (1) Shadows and reflections are currently not modeled by our method and thus won't move with the animation. (2) Since the reconstructed mesh must fit the silhouette, the shape may look unrealistic, e.g., wrong shape of shoes; on the other hand this enables handling abstract art. (3) Our method accounts for self-occlusions when arms partially occlude the head, torso, or legs. It remains future work to handle other occlusions, e.g., legs crossed when sitting. (4) Person detection and segmentation, pose detection and body labeling can fail, requiring manual corrections. (5) We have opted for simple texture inpainting for occluded body parts, with some user interaction if needed. Using deep learning to synthesize, e.g., the appearance of the back of a person given the front, is a promising research area, but current methods that we have tested [38, 21, 39] give very blurry results.

Summary: We have presented a method to create a 3D animation of a person in a single image. Our method works with large variety of of whole-body, fairly frontal photos, ranging from sports photos, to art, and posters. In addition, the user is given the ability to edit the human in the image, view the reconstruction in 3D, and explore it in AR.

We believe the method not only enables new ways for people to enjoy and interact with photos, but also suggests a pathway to reconstructing a virtual avatar from a single image while providing insight into the state of the art of human modeling from a single photo.

Acknowledgements This work was supported by NSF/Intel Visual and Experimental Computing Award #1538618, the UW Reality Lab, Facebook, Google, Huawei, and a Reality Lab Huawei Fellowship. We thank all of the photo owners for allowing us to use their photos; photo credits are given in each figure.

References

[1] Apple corps. 7, 8

[2] Dark-crawler. https://www.deviantart.com/dark-crawler/art/Son-Goku-465455963. 7, 8

[3] Gettyimages. https://www.gettyimages.com/. 7

[4] Kerry varnum photography. https://www.facebook.com/KerryVarnumPhotography. 7, 8

[5] Nikkatsu corp. http://intl.nikkatsu.com/. 7

[6] Superherohype.com. 7

[7] Tabby. http://tabbythis.com/?p=1326. 7

[8] Wikiart: Visual art encyclopedia. https://www.wikiart.org/. 7, 8

[9] Wikimedia. https://commons.wikimedia.org/wiki/Main_Page. 7, 8

[10] T. Alldieck, M. Magnor, W. Xu, C. Theobalt, and G. Pons-Moll. Detailed human avatars from monocular video. In *2018 International Conference on 3D Vision (3DV)*, pages 98–109. IEEE, 2018. 2

[11] T. Alldieck, M. Magnor, W. Xu, C. Theobalt, and G. Pons-Moll. Video based reconstruction of 3d people models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 3, 8

[12] H. Averbuch-Elor, D. Cohen-Or, J. Kopf, and M. F. Cohen. Bringing portraits to life. *ACM Transactions on Graphics (Proceeding of SIGGRAPH Asia 2017)*, 36(4):to appear, 2017. 2

[13] J. Bai, A. Agarwala, M. Agrawala, and R. Ramamoorthi. Automatic cinemagraph portraits. In *Computer Graphics Forum*, volume 32, pages 17–25. Wiley Online Library, 2013. 2

[14] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24–1, 2009. 2, 6

[15] R. Basri, D. Jacobs, and I. Kemelmacher. Photometric stereo with general, unknown lighting. *International Journal of Computer Vision*, 72(3):239–257, 2007. 4

[16] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black. Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image. In *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science. Springer International Publishing, Oct. 2016. 2, 3, 8

[17] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on pattern analysis and machine intelligence*, 23(11):1222–1239, 2001. 5, 6

[18] Y.-Y. Chuang, D. B. Goldman, K. C. Zheng, B. Curless, D. H. Salesin, and R. Szeliski. Animating pictures with stochastic motion textures. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 853–860. ACM, 2005. 2

[19] CMU. *CMU Graphics Lab Motion Capture Database*, 2007. 7

[20] E. Dibra, H. Jain, C. Oztireli, R. Ziegler, and M. Gross. Human shape from silhouettes using generative hks descriptors and cross-modal neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4826–4836, 2017. 2

[21] P. Esser, E. Sutter, and B. Ommer. A variational u-net for conditional appearance and shape generation. *arXiv preprint arXiv:1804.04694*, 2018. 6, 8

[22] M. Flagg, A. Nakazawa, Q. Zhang, S. B. Kang, Y. K. Ryu, I. Essa, and J. M. Rehg. Human video textures. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games*, pages 199–206. ACM, 2009. 2

[23] M. S. Floater. Mean value coordinates. *Computer aided geometric design*, 20(1):19–27, 2003. 4

[24] R. A. Güler, N. Neverova, and I. Kokkinos. Densepose: Dense human pose estimation in the wild. *arXiv preprint arXiv:1802.00434*, 2018. 2

[25] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. *arXiv preprint arXiv:1703.06870*, 2017. 2

[26] A. Hornung, E. Dekkers, and L. Kobbelt. Character animation from 2d pictures and 3d motion data. *ACM Transactions on Graphics (TOG)*, 26(1):1, 2007. 2, 7, 8

[27] A. Jain, T. Thormählen, H.-P. Seidel, and C. Theobalt. Moviereshape: Tracking and reshaping of humans in videos. In *ACM Transactions on Graphics (TOG)*, volume 29, page 148. ACM, 2010. 2

[28] N. Joshi, S. Mehta, S. Drucker, E. Stollnitz, H. Hoppe, M. Uyttendaele, and M. Cohen. Cliplets: juxtaposing still and dynamic imagery. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, pages 251–260. ACM, 2012. 2

[29] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik. End-to-end recovery of human shape and pose. *arXiv preprint arXiv:1712.06584*, 2017. 2

[30] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018. 2

[31] I. Kemelmacher-Shlizerman and S. M. Seitz. Face reconstruction in the wild. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1746–1753. IEEE, 2011. 6

[32] I. Kemelmacher-Shlizerman, E. Shechtman, R. Garg, and S. M. Seitz. Exploring photobios. In *ACM Transactions on Graphics (TOG)*, volume 30, page 61. ACM, 2011. 2

[33] N. Kholgade, T. Simon, A. Efros, and Y. Sheikh. 3d object manipulation in a single photograph using stock 3d models. *ACM Transactions on Graphics (TOG)*, 33(4):127, 2014. 2

[34] D. E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10(Jul):1755–1758, 2009. 6

[35] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in neural information processing systems*, pages 109–117, 2011. 2

[36] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, Oct. 2015. 3

[37] Z. Lun, M. Gadelha, E. Kalogerakis, S. Maji, and R. Wang. 3d shape reconstruction from sketches via multi-view convolutional networks. In *3D Vision (3DV), 2017 International Conference on*, pages 67–77. IEEE, 2017. 2

[38] L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool. Pose guided person image generation. In *Advances in Neural Information Processing Systems*, pages 405–415, 2017. 6, 8

[39] L. Ma, Q. Sun, S. Georgoulis, L. Van Gool, B. Schiele, and M. Fritz. Disentangled person image generation. *arXiv preprint arXiv:1712.02621*, 2017. 6, 8

[40] R. Martin-Brualla, D. Gallup, and S. M. Seitz. Time-lapse mining from internet photos. *ACM Transactions on Graphics (TOG)*, 34(4):62, 2015. 2

[41] Matterport. *Mask R-CNN Implementation by Matterport, Inc*, 2017. 2

[42] M. Omran, C. Lassner, G. Pons-Moll, P. Gehler, and B. Schiele. Neural body fitting: Unifying deep learning and model based human pose and shape estimation. In *2018 International Conference on 3D Vision (3DV)*, pages 484–494. IEEE, 2018. 2

[43] G. Pavlakos, L. Zhu, X. Zhou, and K. Daniilidis. Learning to estimate 3d human pose and shape from a single color image. *arXiv preprint arXiv:1805.04092*, 2018. 2

[44] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. In *ACM Transactions on graphics (TOG)*, volume 22, pages 313–318. ACM, 2003. 6

[45] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM transactions on graphics (TOG)*, volume 23, pages 309–314. ACM, 2004. 6

[46] A. Schödl, R. Szeliski, D. H. Salesin, and I. Essa. Video textures. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 489–498. ACM Press/Addison-Wesley Publishing Co., 2000. 2

[47] J. Tompkin, F. Pece, K. Subr, and J. Kautz. Towards moment imagery: Automatic cinemagraphs. In *Visual Media Production (CVMP), 2011 Conference for*, pages 87–93. IEEE, 2011. 2

[48] G. Varol, D. Ceylan, B. Russell, J. Yang, E. Yumer, I. Laptev, and C. Schmid. BodyNet: Volumetric inference of 3D human body shapes. In *ECCV*, 2018. 2

[49] G. Varol, J. Romero, X. Martin, N. Mahmood, M. Black, I. Laptev, and C. Schmid. Learning from synthetic humans. *arXiv preprint arXiv:1701.01370*, 2017. 2

[50] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016. 2

[51] F. Xu, Y. Liu, C. Stoll, J. Tompkin, G. Bharaj, Q. Dai, H.-P. Seidel, J. Kautz, and C. Theobalt. Video-based characters: creating new human performances from a multi-view video database. *ACM Transactions on Graphics (TOG)*, 30(4):32, 2011. 2

[52] X. Xu, L. Wan, X. Liu, T.-T. Wong, L. Wang, and C.-S. Leung. Animating animal motion from still. In *ACM Transactions on Graphics (TOG)*, volume 27, page 117. ACM, 2008. 2

[53] S. Zhou, H. Fu, L. Liu, D. Cohen-Or, and X. Han. Parametric reshaping of human bodies in images. In *ACM Transactions on Graphics (TOG)*, volume 29, page 126. ACM, 2010. 2

[54] Z. Zhou, B. Shu, S. Zhuo, X. Deng, P. Tan, and S. Lin. Image-based clothes animation for virtual fitting. In *SIGGRAPH Asia 2012 Technical Briefs*, page 33. ACM, 2012. 2

[55] S. Zuffi, A. Kanazawa, and M. J. Black. Lions and tigers and bears: Capturing non-rigid, 3D, articulated shape from images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2018. 2