A. Implementation details

Network architectures. Similar to FOMM [52], for our keypoint predictor C we employ a U-Net [50] backbone operating on 64×64 resolution input images. Following FOMM [52], the architecture consists of five "convolution - batch norm - ReLU - pooling" blocks in the encoder and five "upsample - convolution - batch norm - ReLU" blocks in the decoder. For the generator G, we use a simple decoder architecture. We use embedding size $N_e = 64$ for all the datasets. In order to regularize the embeddings, we apply differentiable whitening [54], which we found to provide additional training stability. Given the embedding $\mathbf{e}$, we transform it to a 4^3 voxel cube with 512 features using a fully connected layer. This voxel cube is then passed through four 3D-Residual Blocks with upsampling. The 3D-Residual block consists of two $3 \times 3 \times 3$ convolutions and two batch normalization in the main branch, and one $1 \times 1 \times 1$ convolution in the residual branch. We choose ReLU for our non-linearity. Each 3D-Residual block reduces the number of features two times, while also increasing the resolution two times using nearest neighbor upsampling. After the final upsampling layer, the voxel cube has size 64^3 and 32 features. We make use of a final projection layer implemented as a $1 \times 1 \times 1$ convolution preceded by batch normalization to obtain $1+3+10$ features for density, RGB, and LBS volumes, respectively.

Neural rendering. For neural rendering, we use fixed camera extrinsics equal to the identity matrix, and a fixed intrinsics matrix assuming a field of view of 0.175 [41]. Based on this configuration, we define the region in which we sample points for rendering to be the cube spanning the region $[-1.0088, 1.0088] \times [-1.0088, 1.0088] \times [9.5000, 11.5000]$. The camera in our settings looks in the positive z direction. We call this region the rendering cube. We use the rendering cube intersections with casted rays to define the near and far camera planes, and uniformly sample 128 points between them. We note that during the $G - phase$, there is no need to capture small parts, as a single part is employed. Thus, to speed up training, we reduce the number of sampling points to 48 in this phase. When mapping the rendering cube to the respective volumes, we increase the rendering cube by a factor of 1.075. For the Cats [79] dataset, we scale the rendering cube by a factor of 1.2. In this manner, we increase the amount of actual space that can be covered by the modeled volume, allowing for a larger set of transformations. Following NeRF [37], we apply perturbations to the sampled point in two ways during training. First, we perturb the position of each point along the ray. Second, we add noise to the sampled densities before rendering. We use a standard deviation of 0.5 for the noise. As the geometry is mostly discovered at the begining of training, we linearly decrease it during each phase of training down to zero at $100k$ training steps.

Perspective-n-Point. We obtain a solution to the PnP problem leveraging the differentiable EPnP [28] implementation from PyTorch3D [44]. For each part p, we predict $N_k = 5^3 = 125$ keypoints, for a total of $N_p \times N_k = 125 \times 10 = 1250$ keypoints. Each keypoint is represented with an unconstrained three-dimensional parameter, followed by sigmoid and normalization to ensure that each individual keypoint always lies inside the rendering cube. We initialize 3D keypoints for each part such that they form a regular cubical grid with 5 equally spaced keypoints on each side of the cube. Due to possible incoherent arrangements between 2D and 3D keypoints, the estimated part poses may correspond to object positions outside the rendering box. This behavior would prevent the affected parts from learning density, since they would bring no contribution to the rendered image. To address this issue, we add an additional loss that pushes the estimated pose of parts with no associated density to the the pose of the largest part (i.e. with the most density). We formulate this loss as follows:

$$\mathcal{L}_{\text{init}} = \sum_p max(0, t - \sigma_p)|T_p - T_{p_{max}}|,$$

where $t = 0.01$ is a density threshold, σ_p is the density associated with part p (which is the mean of all densities for all sampled points, multiplied by the LBS weight) and $T_{p_{max}}$ is the pose of the part with the maximal density. For the first phase, when a single part is learned, we use the identity matrix in place of $T_{p_{max}}$.

Training. In the first phase, we train the model for 200 epochs with a batch size of 128 on 8 A100 GPUs. To equally balance the contribution of each identity, each epoch consists of a single frame randomly sampled from each video in the dataset. In this phase, we decrease the contribution of the $\mathcal{L}_{\text{bkg}}$ by a factor of 0.8 every 10 epochs. For the Cats [79] dataset, the method is trained for 1000 epochs on a single A100 GPU with a batch size of 16. In the second phase, we train the model for 1000 epochs using a batch size of 16 on 8 A100 GPUs. The learning rate is decayed 10 times at 600 epochs and 900 epochs. Finally, to encourage the network to discover small parts such as hands in TEDXPeople [17] dataset, during this phase we also employ co-part segmentation obtained without supervision from MRAA. The loss consists in the cross-entropy loss between MRAA [55] co-parts and our rendered LBS weights. As with $\mathcal{L}_{\text{bkg}}$, this loss is decreased by a factor of 0.8 every 10 epochs. For both phases, we use Adam [27] with $lr = 5e-4$ and $\beta = (0.5, 0.999)$.

Inference. To embed test images, we rely on a two stage procedure, similar to PTI [49]. First, we optimize the embedding $\mathbf{e}$ for a particular image using the reconstruction loss $\mathcal{L}_{\text{r}}$ used during training, and employing Adam [27] with $lr = 1e-2$. The optimization is run for 3000 steps, and the learning rate is decayed by a factor of 10 every

750 steps. Similarly to StyleGAN2 [24], we add random noise to the embedding to promote exploration. We select an initial standard deviation of 0.5 for this noise and decay it until reaching zero at step 1500. For the second stage, we fine-tune all the generator parameters. To avoid forgetting the useful geometry prior learned during training, we add an additional geometry regularization loss:

$$\mathcal{L}_{\text{geo}} = |\hat{V}^{\text{Density}} - V^{\text{Density}}| + |\hat{V}^{\text{LBS}} - V^{\text{LBS}}|,$$

where $\hat{V}^{\text{Density}}$, $\hat{V}^{\text{LBS}}$ are the density and LBS weights from the first stage, respectively. We also employ additional data augmentation at alternate steps by applying random Euclidean transformations for the source image, for which we sample rotation angles and translations in the $[-0.1, 0.1]$ range. Note that, as our model assumes the background is static, we only enforce this loss for the foreground using the rough background mask obtained from the first stage. We train for 500 steps in this stage. For optimization with 5 input frames, we increase the number of steps in the second stage to 3000. Since 5 images may not fit into a single GPU, we use a batch size equal to 2. Inverting one image takes roughly 10 minutes on an A100 GPU, while inverting five images takes roughly 20 minutes. Our PnP-based part pose estimation algorithm introduces some instability in the estimation of the distance of the part from the camera. While this instability does not produce artifacts when rendering the object from limited rotation angles, it becomes more noticeable when rendering from extreme camera angles. To mitigate such effects, we devise an inference-time filtering strategy to smooth abrupt changes in the estimated depth of each part. For each part, we estimate the distance from the camera origin to the center of the rendering cube. We then compute the mean of all these distances for each part l_p. Finally, we rescale the vector from the camera origin to the center of the rendering cube, such that they have the same length l_p in all frames.

B. Dataset details

We employ three training datasets: VoxCeleb [38], TEDXPeople [17] and Cats [79]. We adopt the VoxCeleb [38] preprocessing of FOMM [52], and preprocess Cats [79] in the same way as [10, 58]. For TEDXPeople [17], we first download the videos listed in [17], then, using the provided timestamps, select continuous chunks of videos starting at the provided timestamp and lasting at most 512 frames. In each chunk, we detect human keypoints and bounding boxes for each frame using [69]. We clamp the predicted bounding box at the hip joints at the bottom of the frame, then increase its size by a factor of 1.2 so as to capture the subject's full upper body, then make it square. We process the video chunk frame-by-frame, adding each processed frame to the current video sample. If, in some frames, the human is not detected or the bounding box moves significantly from the initial position, we stop the current video sample and start collecting a new sample at the next detection of a human. To further clean the dataset, we discard video samples that are too short (less than 64 frames), to small (less than 256 pixels on any side), have significant background movement (detected using simple $\mathcal{L}_1$ error on pixel values), have no movement in foreground (which most likely indicate that the detected human is a static image visualized during the presentation) or have a width similar to the height (which indicates a failure of the hip predictor). We select only views marked as "front" in the original annotations [17], and from each YouTube video, we take at most three different samples. Our final dataset consists of 40896 different samples from 17451 different YouTube videos.

C. Metric details

A critical aspect of 3D animation is the ability to synthesize novel views of the observed target object. However, evaluating this ability is challenging, as animation datasets typically lack multi-view observations. We thus introduce metrics that, given a triplet composed of a source frame, a driving frame, and a result rendered under a target camera, can quantitatively evaluate the quality of the rendered novel view:

- Average Yaw Deviation (AYD): this evaluates whether the object is rendered from the target camera perspective. Given the yaw angle between the camera and the object in the driving frame Θ_d and in the rendered frame Θ_r, and the yaw angle of the novel view camera with respect to the original camera Θ_c, we define $\text{AYD} = |\Theta_d - (\Theta_c + \Theta_r)|$
- Average Shape Consistency (ASC): this evaluates whether the identity of the rendered object is the one in the source frame. Given an identity code for the source frame $c_s^s \in \mathcal{R}^{N_{c^s}}$ and an identity code for the rendered frame c_r^s, we define $\text{ASC} = \frac{|c_s^s - c_r^s|}{N_{c^s}}$
- Average Pose Consistency (APC): this evaluates whether the object is rendered in the pose given by the driving frame. Given a pose code for the driving frame c_d^p and a pose code for the rendered frame $c_r^p \in \mathcal{R}^{N_{c^p}}$, we define $\text{APC} = \frac{|c_d^p - c_r^p|}{N_{c^p}}$

We obtain Θ, c^s and c^p in a way that is specific to the given object category. For faces, we compute the head yaw angle Θ using the 6DOF head pose estimator 6DRepNet [19], which we find robust to extreme head poses and possible corrupted regions in the rendered frames. Given an image, the model directly provides the estimated yaw angle, which we convert to radians prior to the computation. To compute c^s and c^p we use the DECA [13] 3DMM. We select this

model due to its robustness to large head rotations, its fast, encoder-based inference, and its ability to disentangle the head shape from the current expression. In particular, we define c^s as the inferred $N_{c^s} = 100$ FLAME [31] face shape parameter, which encodes the identity of the subject. We define c^p as the concatenation of the inferred 50 expression parameters with the estimated jaw rotation in axis-angle representation for $N_{c^p} = 53$, which together capture the particular facial pose. We choose not to make use of the estimated head yaw angle, since we find it less robust than the one inferred from our adopted 6DOF head pose estimator. For human bodies, we fit the SMPL [34] body model to each frame using 3DCrowdNet [9]. We choose 3DCrowdNet due to its fast inference time and its robustness to partially-occluded subjects which are frequent in the TEDXPeople [17] dataset, where only the upper half of the body is typically present in the frame. 3DCrowdNet requires a set of 2D human body keypoints to be detected for each frame. We first detect person bounding boxes using Faster R-CNN [47] and use VitPose [71] to detect the 2D human body keypoints, which we find to work robustly even in the presence of artifacts in the images. Given the fitted SMPL model, we define c^s as the inferred $N_{c^s} = 10$ body shape parameters, and c^p as the concatenation of the inferred angles for a selected set of 13 joints in axis-angle representation corresponding to the joints situated above the 'belly button' joint for a total of $c^p = 39$ elements. Selection of the joints ensures that joints that are typically not present in the TEDXPeople dataset, and thus cannot be reliably estimated, will not negatively affect the precision of the evaluation. We extract the yaw angle Θ by transforming the root joint axis angle rotation inferred by the model into the corresponding rotation matrix $M = M_y M_x M_z$, and extract the yaw angle Θ of the M_y component representing the y-axis rotation matrix as follows:

$$\begin{aligned} \text{pitch} &= \arcsin M_{1,2} \\ \cos(\Theta) &= \frac{M_{2,2}}{\cos(\text{pitch})} \\ \sin(\Theta) &= \frac{M_{0,2}}{\cos(\text{pitch})} \\ \Theta &= \arctan_2(\sin(\Theta), \cos(\Theta)), \end{aligned}$$

where the case of $\cos(\text{pitch}) = 0$ is disregarded, since in practice we never render objects from high-pitch angles.

We now define the evaluation protocols followed for the animation and novel view synthesis tasks. For the animation task, we consider each test set video and select the first frame of each video as the source frame. We consider as driving frames five video frames, equally spaced along the duration of the test video. We then generate the object in the source frame in the pose of each driving frame under novel views, produced by rotating the object with the following Θ angles: $0, \pm\frac{\pi}{12}, \pm\frac{\pi}{6}, \pm\frac{\pi}{4}$. The triplets built from all combinations of source, driving and rendered frames are used for the computation of AYD, ASC and APC. For the novel view synthesis, we consider as source and driving frame the same, first frame of each video. This allows pure evaluation of the novel view synthesis capabilities of the method. We render each frame under the set of 256 linearly sampled Θ angles in the range $[-\frac{\pi}{2}, +\frac{\pi}{2}]$ and compute AYD, ASC and APC using all the available frame triplets.

D. Baseline details

MRAA and FOMM. MRAA and FOMM rely on affine transformations to transfer motion. Thus, in order to perform novel view synthesis a natural idea would be to modify these affine transformations such that they represent the object in the novel view. We achieve this with the following procedure. First, near each region center, we sample 4 additional keypoints in a small distance $= 0.05$ forming a cross centered around the central keypoint for the region. The region center and these additional keypoints are then lifted to the 3D space using a depth map obtained from the driving image with off-the-shelf depth estimator [12]. As we need to recover depth for the object in the pose of the frame for which to perform novel view synthesis, we use absolute depth for animation to ensure the rendered frame pose is the same as the driving frame. These points are then projected to the target view using the desired camera parameters. Finally, we estimate a new affine transformation from these projected points. Note the off-the-shelf depth estimator only provides relative depth, and it is thus not possible to utilize it directly. To overcome this issue, we leverage depth obtained from our method and find a linear mapping between our depth and off-the-shelf depth. This linear mapping is consists of d_{scale} and d_{shift} parameters and can be find in closed form:

$$d_{scale} = \frac{Cov(d, \hat{d})}{Var(\hat{d})},$$

$$d_{shift} = E[d] - d_{scale} E[\hat{d}],$$

where d is the depth map from our method, $\hat{d}$ is the off-the-shelf depth map, E is sample mean, Var is sample variance, and Cov is sample covariance.

LIA. LIA expresses animation as navigation inside a learned latent space. Given an embedding z_s in this latent space for the source image, animation is expressed as $z_d = z_s + w = z_s + \sum a_i d_i$, with z_d expressing the latent code corresponding to the animated result and vector w expressed as the summation of a set of learned motion directions d multiplied by corresponding magnitudes a, which form an orthogonal basis of the latent space. The set of learned motion directions represents the main types of motions performed by the objects. Interestingly, we find that for the VoxCeleb dataset, d_2, the second of such

	VoxCeleb			TEDXPeople		
Method	AYD↓	ASC↓	APC↓	AYD↓	ASC↓	APC↓
FOMM [52]	0.801	0.145	0.194	0.639	0.029	1.14
MRAA [55]	0.760	0.133	0.177	0.686	**0.022**	**0.861**
LIA [64]	0.188	0.132	0.198	-	-	-
Our 1 frame	0.155	**0.119**	**0.171**	0.248	0.023	0.941
Our 5 frame	**0.153**	0.126	0.184	**0.244**	0.024	0.959

Table 3. The results of generating novel views of the first frame of each video sequence. Camera angles range from $-90°$ to $+90°$.

directions, is correlated with y-axis head rotation. While this movement is undesirably entangled with other motion components such as x-axis head rotation, we exploit this finding to produce novel views. Since no immediate correspondence between magnitude a_2 added to such direction and Θ exists a priori, we build a linear model mapping changes in Θ between the source and driving frame with a_2. To build such a linear model, we consider the first frame of each video and produce novel views using values of a_2 in the range of $[-17, +17]$ degrees. For each generated novel view, we evaluate the corresponding changes in Θ between the source and driving frame using 6DRepNet [19] and use such data to fit our linear model $a_2 = 7.453\Theta$. Given a desired Θ angle, we leverage the linear model to devise the magnitude a_2 and produce $z_{\text{novel}} = z_d + a_2 d_2$, which is decoded to the frame under the novel view. Note that since the linear model directly optimizes the Θ error on the test set, we expect the AYD metric produced for such baseline to be biased toward lower values.

E. Novel view synthesis

In this section, we evaluate the capabilities of the animation methods to perform novel view synthesis without animation. To this end, we simply rotate the image along the y axis on the set of angles from $-90°$ to $+90°$. The results are provided in Tab. 3, and confirm the findings from Sec. 4.2. While MRAA has favourable ASC and APC errors, it has very high AYD. This behavior is expected, because the method is simply performing a translation of the subject, rather than rotating it according to the provided yaw angles. This ensures the pose and identity remain preserved, at the cost of performing poor novel view synthesis. LIA, on the other hand, has a low AYD, while ASC and APC are high, which confirms that LIA has entangled latent directions that prevent novel view synthesis without significantly altering the pose and identity. Our model achieves the best AYD, which suggests that it performs the most accurate camera manipulations.

F. Canonical visualization

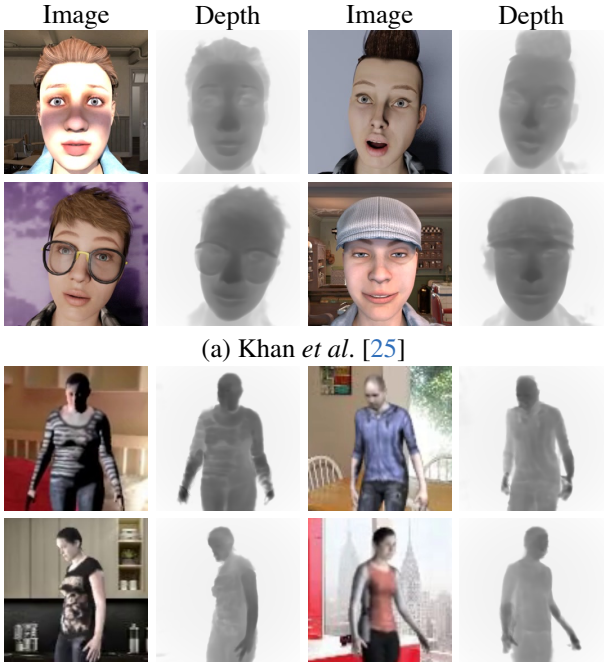
In order to better demonstrate the representation learned by our model, we visualize some of the training identities in the canonical pose, i.e. where T_p is the identity matrix

(a) VoxCeleb [38] (b) TEDXPeople [17]

Figure 6. Visualization of canonical spaces.

for all parts. Note that, since there are no prior assumptions on how the object should be placed in the rendering cube, parts seen from the camera with identity matrix extrinsics may be arbitrary. Thus, we select the camera from which objects will look reasonable. Note that T_p is still the same for all parts and all objects. The visual results are presented in Fig 6, which clearly shows that all objects have the same pose, which is a crucial property for animation.

G. Synthetic



(a) Khan *et al*. [25]

(b) SURREAL [60]

Figure 7. Visualization of predicted depth for synthetic datasets. We show the input image and the depth predicted by our method.

To further evaluate the quality of the learned geometry, we ran experiments on images from two synthetically rendered datasets providing ground truth depth: 1.) that of Khan *et al*. [25], which provides high-quality, portrait-style facial images; and 2.) SURREAL [60], which provides full-body renderings of animated subjects. We use 112

image for faces and 60 images for bodies, cropped such that they roughly correspond to the cropping used in the respective real datasets used for training. These datasets contain subjects with widely varying identities, poses, hairstyles, and attire, rendered in different environments and lighting conditions. However, for these experiments we did not rely on synthetic data for training, instead using models pretrained on 2D images from VoxCeleb [38] or TEDXPeople [17] for faces or bodies, respectively. Despite the domain gap between our training and evaluation data, we are able to obtain high-quality depth estimates for these synthetic renderings using models trained only on real, in-the-wild images. Given a synthetic input image, we invert it, then compute the Pearson correlation coefficient between our method's inferred depth and the ground truth. For these experiments, as we are only concerned with the geometry of the target object, we masked out the depth for background regions, computing the correlation only between the depths of foreground pixels. We compare our predicted depth with the general purpose state-of-the-art depth predictor Omnidata [12]. The depth correlation for Omnidata is 0.602 for faces and 0.470 for bodies, while for our method they are 0.793 and 0.568, respectively. In Fig. 7, we show the image along with the reconstructed depth. These results demonstrate that our unsupervised method learns meaningful geometric representations, even for significantly out-of-distribution inference data.

H. Limitations

Our model addresses, for the first time, the task of unsupervised 3D animation. While our model obtains compelling results on this challenging task, here we note some limitations:

- Our method assumes the object can be represented with a voxel cube of size 64^3. We notice that when generating novel views involving large camera displacements from the original pose, some seam-like artifacts may appear. We believe they are due to the small size of the voxel cube and errors in predicting precisely the distance of the part, which could lead to a slight displacement between different parts.

- For each test identity, our model makes use of an optimization-based procedure to compute the respective identity embedding and fine-tune the generator. This procedure increases the inference cost of our model, but needs to be performed only once for each test identity, thus the cost of the procedure is amortized when producing a large number of frames.

- Our model renders frames at a resolution of 256×256, which is lower than the ones typically supported by state of the art 2D animation methods. This is a common limitation of 3D methods based on volumetric rendering, and we expect continuous progress in efficient volumetric representations and rendering to enable the generation of higher resolution images.

- Our method can learn geometry only from the views that were observed in the training dataset, thus, for the back side of the face in VoxCeleb [38] and the back side of the body in TEDXPeople [17], no precise geometry is learned.

I. Failed Experiments

Canonical space representation. During our initial experiments we tried many different representations for canonical space: Triplanar [5], MLP [37], CP and VM decomposed cubes [7]. However, we found that decomposed solutions such as Triplanar [5] and VM [7] are biased towards flat geometry, while an MLP [37] is extremely slow. Note that the Triplanar [5] representation also utilizes a small MLP, thus in our experiments it was slower than directly sampling our Voxel cube.

Pose prediction. Before reaching the PnP formulation, we tried many different approaches for pose prediction. First, we started with *Direct* approaches, and we tested several architectures and rotation representations [83]. However, all of them failed to produce meaningful geometry. We also tried an optimization-based approach for motion, i.e. having a pose parameter for each frame in the dataset. While this produced decent results for a single video, when the number of frames scales to millions, this approach quickly becomes infeasible.

Different PnP. We tested several different PnP implementations. We found that implementations based on declarative layers [8,16] are extremely slow, and using them in our setting would have been unfeasible. We also tested an implementation from the Kornia Library [48] that is based on DLT. However, it did not produce any meaningful results and produced divergence of the model. Our final choice was the EPnP [28] implementation from Pytorch3D [44]. However, we would like to note that it was only working in PyTorch 10.1 and not PyTorch 11, where it was not converging. We discovered the problem was the initially unstable gradients of the *pinverse* function in the newer version. We think that this instability can be solved with better initialization for the 2d points, however we left this investigation for future work.

Depth and normal supervision. To help discovering the geometry we also tried to utilize depth and normal supervision from an off-the-shelf predictor [12]. Note that, because the normals supervision require computation of second order derivatives and we rely on voxel sampling with grid_sample, we need a second derivative of grid_sample,

which is not implemented in PyTorch[3]. Thus, we develop a custom cuda kernel for the second derivative. While the depth and normal supervision helps to improve results for one-phase training, we found it to be unnecessary with two-phase training.

Upsampler. We render images in full resolution, however in prior experiments we utilize an upsampler. While this method works faster and consumes less memory, it produces less detailed geometry and worse view consistency.

Different multipart representations. We also tried two different representations for describing objects with multiple parts. The first had a shared radiance V^{RGB} volume, but a separate density for each part, while the second used different radiance and density volume for each part. Both of these strategies produce reasonable results. However, for them it is much harder to discover a large number of parts, and they usually degrade to solutions with only one or two parts being used.

Few shot NeRF regularization. We also tested several few-shot NeRF regularization techniques: entropy loss on the NeRF weights [26], loss on the weights from MiP NeRF [2], surface normals regularization [61] and warping loss from MVCGAN [80]. We found that all of them are unnecessary with our two phase training strategy.

Discriminator. To regularize the novel views, we also try to employ a Discriminator, similarly to 3D-GANs [5, 58]. In more detail, we first predict the pose of the object and then try to rotate this pose to generate the object in the novel view. This image is subsequently passed to the discriminator. However, we found it hard to find proper rotation ranges, thus the discriminator reduced the quality of the geometry in our experiments.

J. Ethical considerations

Dataset usage. The primary datasets used in our experiments, VoxCeleb [38] and TEDXPeople [17], contain publicly available videos of notable figures in public venues, *e.g*. celebrities giving interviews and speakers giving presentations to large audiences. These datasets have been released by and employed for prior academic research, *e.g*. the works we use for our comparisons and evaluations.

Other datasets, such as Khan *et al*. [25] and SURREAL [60] which are used for our ground-truth depth inference evaluations, contain realistic but synthetic images rendered from 3D models of human faces and bodies, respectively. SURREAL [60] uses body scans and motion capture sequences generated from 3D capture of the appearances and performances of subjects who consented to have this captured and released for academic purposes. Khan *et al*. [25] contains facial images generated by perturbing characteristics such as facial identity, hair and clothing for models in a standard 3D modeling and rendering framework, and thus do not correspond to any particular person whose identity may be at risk of being revealed. Each of these datasets are both publicly available and have been used for prior academic works. As such, there are no particular concerns about violating the privacy or anonymity of our test subjects.

Potential for bias in synthesis results. As with other data-driven methods for performance-driven animation, the amount of variation in characteristics such as gender, age, body type, and ethnicity that can be handled by our methods with the source and driving subjects while producing plausible synthesis results is dependent on the amount of such variations contained in the dataset. While the variations in the real and synthetic images used in our experiments are limited by those in the aforementioned datasets used in our evaluations, *e.g*. in typical celebrity videos and TEDx presentations, our method has no particular limitations towards such subjects, and thus could be deployed on other datasets containing different identity characteristics. Deploying this approach in a manner which is fair and robust with respect to such variations for non-academic purposes, such as commercial applications, would require employing a dataset that is appropriately representative of the possible target identities, and evaluating the results to ensure consistent behavior across these demographics. However, for the academic evaluations presented here, our evaluations suggest that our approach works as expected given the datasets we use, and thus could generalize to other training datasets fairly easily. Finally, as our approach only relies on unconstrained video sequences for training, acquiring the data needed to adapt to new subjects is fairly straightforward, provided that the appropriate video sequences can be collected for training. As such, there are no particular concerns related to unfair bias in our approach.

Possibly misuse. As with other works in the domain of realistic, performance-driven animation, our work carries with it the possibility of use for deceptive activities, *e.g*., creating plausible videos of public figures as misinformation to advance a political agenda. However, we maintain that, while this is clearly a valid concern for the near future, developing and studying such technology in public forms such as this work raises awareness of this potential, and with it the skepticism of viewers towards potentially misleading videos. Furthermore, publicly describing our work and results allows for the advancement of forensic methods to identify when such manipulations have occurred. We thus believe that our work helps to prevent the secretive development and deployment of these techniques for malicious ends which are not known or detectable either to average media consumers or professional forensic analysts.

[3] https://github.com/pytorch/pytorch/issues/34704

References

[1] Sherwin Bahmani, Jeong Joon Park, Despoina Paschalidou, Hao Tang, Gordon Wetzstein, Leonidas Guibas, Luc Van Gool, and Radu Timofte. 3d-aware video generation. *arXiv:2206.14797*, 2022. 2

[2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 14

[3] Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. Optimizing the latent space of generative networks. *arXiv preprint arXiv:1707.05776*, 2017. 3

[4] Caroline Chan, Shiry Ginosar, Tinghui Zhou, and Alexei A Efros. Everybody dance now. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 3

[5] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2, 13, 14

[6] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2

[7] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. *arXiv preprint arXiv:2203.09517*, 2022. 13

[8] Bo Chen, Alvaro Parra, Jiewei Cao, Nan Li, and Tat-Jun Chin. End-to-end learnable geometric vision by backpropagating pnp optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 4, 13

[9] Hongsuk Choi, Gyeongsik Moon, JoonKyu Park, and Kyoung Mu Lee. Learning to estimate robust 3d human mesh from in-the-wild crowded scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 6, 11

[10] Yu Deng, Jiaolong Yang, Jianfeng Xiang, and Xin Tong. Gram: Generative radiance manifolds for 3d-aware image generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2, 10

[11] Nikita Drobyshev, Jenya Chelishev, Taras Khakhulin, Aleksei Ivakhnenko, Victor Lempitsky, and Egor Zakharov. Megaportraits: One-shot megapixel neural head avatars. In *Proceedings of the ACM International Conference on Multimedia*, 2022. 3

[12] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 6, 11, 13

[13] Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an animatable detailed 3D face model from in-the-wild images. *ACM Transactions on Graphics*, 2021. 6, 10

[14] Zhenglin Geng, Chen Cao, and Sergey Tulyakov. 3d guided fine-grained face manipulation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 3

[15] Zhenglin Geng, Chen Cao, and Sergey Tulyakov. Towards photo-realistic facial expression manipulation. *International Journal of Computer Vision*, 2020. 3

[16] Stephen Gould, Richard Hartley, and Dylan Campbell. Deep declarative networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 4, 13

[17] Artur Grigorev, Karim Iskakov, Anastasia Ianina, Renat Bashirov, Ilya Zakharkin, Alexander Vakhitov, and Victor Lempitsky. Stylepeople: A generative model of fullbody human avatars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 6, 9, 10, 11, 12, 13, 14

[18] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d aware generator for high-resolution image synthesis. In *Proceedings of the International Conference on Machine Learning*, 2022. 2

[19] Thorsten Hempel, Ahmed A. Abdelrahman, and Ayoub Al-Hamadi. 6d rotation representation for unconstrained head pose estimation. In *Proceedings of the IEEE International Conference on Image Processing*, 2022. 10, 12

[20] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Unsupervised learning of object landmarks through conditional image generation. In *Proceedings of the Neural Information Processing Systems Conference*, 2018. 4

[21] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. Neuman: Neural human radiance field from a single video. In *Proceedings of the European Conference on Computer Vision*, 2022. 3

[22] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of the European Conference on Computer Vision*, 2016. 5

[23] Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision*, 2018. 2

[24] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 3, 10

[25] Faisal Khan, Shahid Hussain, Shubhajit Basak, Joseph Lemley, and Peter Corcoran. An efficient encoder–decoder model for portrait depth estimation from single images trained on pixel-accurate synthetic data. *Neural Networks*, 2021. 12, 14

[26] Mijeong Kim, Seonguk Seo, and Bohyung Han. Infonerf: Ray entropy minimization for few-shot neural volume rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 14

[27] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 9

[28] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision*, 2009. 3, 4, 9, 13

[29] J. P. Lewis, Matt Cordner, and Nickson Fong. Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In *Special Interest Group on Computer Graphics and Interactive Techniques*, 2000. 3, 4

[30] Ruilong Li, Julian Tanke, Minh Vo, Michael Zollhofer, Jurgen Gall, Angjoo Kanazawa, and Christoph Lassner. Tava: Template-free animatable volumetric actors. In *Proceedings of the European Conference on Computer Vision*, 2022. 5

[31] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *Special Interest Group on Computer Graphics and Interactive Techniques*, 2017. 11

[32] Xueting Li, Sifei Liu, Shalini De Mello, Kihwan Kim, Xiaolong Wang, Ming-Hsuan Yang, and Jan Kautz. Online adaptation for consistent mesh reconstruction in the wild. In *Proceedings of the Neural Information Processing Systems Conference*, 2020. 2, 3

[33] Wen Liu, Zhixin Piao, Jie Min, Wenhan Luo, Lin Ma, and Shenghua Gao. Liquid warping gan: A unified framework for human motion imitation, appearance transfer and novel view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 3

[34] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *Special Interest Group on Computer Graphics and Interactive Techniques*, 2015. 11

[35] Dominik Lorenz, Leonard Bereska, Timo Milbich, and Bjorn Ommer. Unsupervised part-based disentangling of object shape and appearance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 3

[36] Arun Mallya, Ting-Chun Wang, and Ming-Yu Liu. Implicit warping for animation with image sets. In *Proceedings of the Neural Information Processing Systems Conference*, 2022. 3

[37] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision*, 2020. 2, 3, 5, 9, 13

[38] Arsha Nagrani, Joon Son Chung, Weidi Xie, and Andrew Zisserman. Voxceleb: Large-scale speaker verification in the wild. *Computer Science and Language*, 2019. 2, 6, 7, 10, 12, 13, 14

[39] Thu H Nguyen-Phuoc, Christian Richardt, Long Mai, Yongliang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. In *Proceedings of the Neural Information Processing Systems Conference*, 2020. 2

[40] Michael Niemeyer and Andreas Geiger. Campari: Camera-aware decomposed generative neural radiance fields. In *Proceedings of the International Conference on 3D Vision*, 2021. 2

[41] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 9

[42] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. StyleSDF: High-Resolution 3D-Consistent Image and Geometry Generation. *arXiv:2112.11427*, 2021. 2

[43] Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. Animatable neural radiance fields for modeling dynamic human bodies. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 3

[44] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 4, 9, 13

[45] Daniel Rebain, Mark Matthews, Kwang Moo Yi, Dmitry Lagun, and Andrea Tagliasacchi. Lolnerf: Learn from one look. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2

[46] Jian Ren, Menglei Chai, Oliver J Woodford, Kyle Olszewski, and Sergey Tulyakov. Flow guided transformable bottleneck networks for motion retargeting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 3

[47] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proceedings of the Neural Information Processing Systems Conference*, 2015. 11

[48] Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. Kornia: an open source differentiable computer vision library for pytorch. In *Proceedings of the Winter Conference on Applications of Computer Vision*, 2020. 13

[49] Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *ACM Transactions on Graphics*, 2022. 6, 9

[50] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer Assisted Intervention*, 2015. 9

[51] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Proceedings of the Neural Information Processing Systems Conference*, 2020. 2

[52] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. In *Proceedings of the Neural Information Processing Systems Conference*, 2019. 2, 3, 5, 6, 7, 8, 9, 10, 12

[53] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. Animating arbitrary objects via deep motion transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2, 3, 7

[54] Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening and coloring transform for GANs. In *Proceedings*

of the International Conference on Machine Learning, 2019. 9

[55] Aliaksandr Siarohin, Oliver Woodford, Jian Ren, Menglei Chai, and Sergey Tulyakov. Motion representations for articulated animation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 3, 4, 5, 6, 7, 8, 9, 12

[56] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014. 5

[57] Ivan Skorokhodov, Sergey Tulyakov, and Mohamed Elhoseiny. Stylegan-v: A continuous video generator with the price, image quality and perks of stylegan2. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2

[58] Ivan Skorokhodov, Sergey Tulyakov, Yiqun Wang, and Peter Wonka. Epigraf: Rethinking training of 3d gans. In *Proceedings of the Neural Information Processing Systems Conference*, 2022. 2, 6, 7, 10, 14

[59] Jiale Tao, Biao Wang, Borun Xu, Tiezheng Ge, Yuning Jiang, Wen Li, and Lixin Duan. Structure-aware motion transfer with deformable anchor model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2, 3

[60] Gül Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 12, 14

[61] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 14

[62] Ting-Chun Wang, Ming-Yu Liu, Andrew Tao, Guilin Liu, Jan Kautz, and Bryan Catanzaro. Few-shot video-to-video synthesis. In *Proceedings of the Neural Information Processing Systems Conference*, 2019. 3

[63] Ting-Chun Wang, Arun Mallya, and Ming-Yu Liu. One-shot free-view neural talking-head synthesis for video conferencing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 3

[64] Yaohui Wang, Di Yang, Francois Bremond, and Antitza Dantcheva. Latent image animator: Learning to animate images via latent space navigation. In *Proceedings of the International Conference on Machine Learning*, 2022. 2, 3, 6, 7, 8, 12

[65] Chung-Yi Weng, Brian Curless, Pratul P Srinivasan, Jonathan T Barron, and Ira Kemelmacher-Shlizerman. Humannerf: Free-viewpoint rendering of moving people from monocular video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 3, 5

[66] Olivia Wiles, A Koepke, and Andrew Zisserman. X2face: A network for controlling face generation using images, audio, and pose codes. In *Proceedings of the European Conference on Computer Vision*, 2018. 3

[67] Shangzhe Wu, Tomas Jakab, Christian Rupprecht, and Andrea Vedaldi. DOVE: Learning deformable 3d objects by watching videos. *arXiv preprint arXiv:2107.10844*, 2021. 2, 3

[68] Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. Unsupervised learning of probably symmetric deformable 3d objects from images in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2

[69] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 10

[70] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 2

[71] Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. Vitpose: Simple vision transformer baselines for human pose estimation. *arXiv preprint arXiv:2204.12484*, 2022. 11

[72] Yang Xue, Yuheng Li, Krishna Kumar Singh, and Yong Jae Lee. Giraffe hd: A high-resolution 3d-aware generative model. *arXiv:2203.14954*, 2022. 2

[73] Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Huiwen Chang, Deva Ramanan, William T Freeman, and Ce Liu. Lasr: Learning articulated shape reconstruction from a monocular video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 3

[74] Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Ce Liu, and Deva Ramanan. Viser: Video-specific surface embeddings for articulated 3d shape reconstruction. In *Proceedings of the Neural Information Processing Systems Conference*, 2021. 2, 3

[75] Gengshan Yang, Minh Vo, Natalia Neverova, Deva Ramanan, Andrea Vedaldi, and Hanbyul Joo. Banmo: Building animatable 3d neural models from many casual videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2, 3

[76] Sihyun Yu, Jihoon Tack, Sangwoo Mo, Hyunsu Kim, Junho Kim, Jung-Woo Ha, and Jinwoo Shin. Generating videos with dynamics-aware implicit generative adversarial networks. In *Proceedings of the International Conference on Machine Learning*, 2022. 2

[77] Egor Zakharov, Aleksei Ivakhnenko, Aliaksandra Shysheya, and Victor Lempitsky. Fast bi-layer neural synthesis of one-shot realistic head avatars. In *Proceedings of the European Conference on Computer Vision*, 2020. 3

[78] Egor Zakharov, Aliaksandra Shysheya, Egor Burkov, and Victor Lempitsky. Few-shot adversarial learning of realistic neural talking head models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 3

[79] Weiwei Zhang, Jian Sun, and Xiaoou Tang. Cat head detection-how to effectively exploit shape and texture features. In *Proceedings of the European Conference on Computer Vision*, 2008. 2, 6, 9, 10

[80] Xuanmeng Zhang, Zhedong Zheng, Daiheng Gao, Bang Zhang, Pan Pan, and Yi Yang. Multi-view consistent generative adversarial networks for 3d-aware image synthesis.

In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2, 14

[81] Jian Zhao and Hui Zhang. Thin-plate spline motion model for image animation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2, 3

[82] Xiaoming Zhao, Fangchang Ma, David Güera, Zhile Ren, Alexander G. Schwing, and Alex Colburn. Generative multiplane images: Making a 2d gan 3d-aware. In *Proceedings of the European Conference on Computer Vision*, 2022. 2

[83] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 8, 13