A First Order Motion Model

We built our architecture on top of First Order Motion Model (Siarohin et al., 2019), a state-of-the-art model on image animation. This section summarizes its background. First Order Motion Model consists of a motion estimation module and an image generation module. The motion estimation module takes as inputs a source image $\mathbf{S} \in \mathbb{R}^{3\times H\times W}$ and a driving image $\mathbf{D} \in \mathbb{R}^{3\times H\times W}$, and predicts a dense motion field $\hat{\mathcal{T}}_{\mathbf{S}\leftarrow\mathbf{D}}$ and an occlusion mask $\hat{\mathcal{O}}_{\mathbf{S}\leftarrow\mathbf{D}}$. The image generation module warps the source image based on the dense motion field $\hat{\mathcal{T}}_{\mathbf{S}\leftarrow\mathbf{D}}$ and inpaints the occluded parts of the source image.

A.1 Motion estimation module

The motion estimation module first uses local affine transformations to approximate motion in the neighborhood of each keypoint. It then combines the local motions to estimate the final dense motion field $\hat{\mathcal{T}}_{\mathbf{S}\leftarrow\mathbf{D}}$.

Approximating motion description Given a driving frame $\mathbf{D}$ and a source frame $\mathbf{S}$, the motion estimation module estimates the backward optical flow $\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{D}}$ from $\mathbf{D}$ to $\mathbf{S}$. The estimation of $\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{D}}$ is based on an abstract reference frame $\mathbf{R}$. Therefore, the estimation of $\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{D}}$ is a combination of estimating $\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}}$ and $\mathcal{T}_{\mathbf{R}\leftarrow\mathbf{D}}$. To estimate the transformation $\mathcal{T}_{\mathbf{X}\leftarrow\mathbf{R}}$ of a given frame $\mathbf{X}$, the motion estimation module considers its first-order Taylor expansions in K keypoints $p_1, \dots, p_K$ in the reference frame $\mathbf{R}$. In the rest of the section, p denotes the point locations in the reference pose space and z denotes the point locations in the $\mathbf{X}$, $\mathbf{S}$, or $\mathbf{D}$ pose spaces.

$$\mathcal{T}_{\mathbf{X}\leftarrow\mathbf{R}}(p) = \mathcal{T}_{\mathbf{X}\leftarrow\mathbf{R}}(p_k) + \left(\frac{d}{dp}\mathcal{T}_{\mathbf{X}\leftarrow\mathbf{R}}(p) \;\middle|\; p = p_k\right)(p - p_k) + o(\|p - p_k\|) \tag{1}$$

Finally, the backward optical flow $\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{D}}$ is obtained as follows:

$$\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{D}} = \mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}} \circ \mathcal{T}_{\mathbf{R}\leftarrow\mathbf{D}} = \mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}} \circ \mathcal{T}^{-1}_{\mathbf{D}\leftarrow\mathbf{R}} \tag{2}$$

Computing the first-order Taylor expansion of Equation 2 gives:

$$\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{D}}(z) \approx \mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}}(p_k) + J_k(z - \mathcal{T}_{\mathbf{D}\leftarrow\mathbf{R}}(p_k)) \tag{3}$$

where J_k is the Jacobian corresponding to the keypoint location p_k in $\mathbf{R}$. To estimate the transformation $\mathcal{T}_{\mathbf{X}\leftarrow\mathbf{R}}$ near p_k in the reference frame $\mathbf{R}$, the motion estimation module uses the standard U-Net architecture as a keypoint detector to estimate a heatmap for each keypoint p_k. The keypoint detector uses softmax activations as a confidence map in the last layer of the decoder for prediction.

Combining local motions The motion estimation module uses a convolutional network to estimate the final dense motion field $\hat{\mathcal{T}}_{\mathbf{S}\leftarrow\mathbf{D}}$ from the set of Taylor approximations of $\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{D}}(z)$. Using the local transformation estimated in Equation 3, it obtains K transformed images $\mathbf{S}^1, \dots, \mathbf{S}^K$. Each image $\mathbf{S}^k$ is aligned with $\hat{\mathcal{T}}_{\mathbf{S}\leftarrow\mathbf{D}}$ in the neighborhood of a keypoint. It obtains also an image $\mathbf{S}^0$ for the background.

To indicate to the dense motion network where each transformation occurs, it further computes heatmaps $\mathbf{H}_k$ for each keypoint p_k. Each $\mathbf{H}_k(z)$ is the difference between the heatmaps centered in $\mathcal{T}_{\mathbf{D}\leftarrow\mathbf{R}}(p_k)$ and $\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}}(p_k)$:

$$\mathbf{H}_k(z) = \exp\left(\frac{(\mathcal{T}_{\mathbf{D}\leftarrow\mathbf{R}}(p_k) - z)^2}{\sigma}\right) - \exp\left(\frac{(\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}}(p_k) - z)^2}{\sigma}\right) \tag{4}$$

where σ is a small fixed standard deviation. By concatenating the heatmaps $\mathbf{H}_k$ and the transformed images $\mathbf{S}^0, \dots, \mathbf{S}^K$, it estimates $K + 1$ masks $\mathbf{M}_k$ that indicate where each local transformation holds. The final dense motion prediction is obtained as follows:

$$\hat{\mathcal{T}}_{\mathbf{S}\leftarrow\mathbf{D}}(z) = \mathbf{M}_0 z + \sum_{k=1}^{K} \mathbf{M}_k(\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}}(p_k) + J_k(z - \mathcal{T}_{\mathbf{D}\leftarrow\mathbf{R}}(p_k))) \tag{5}$$

(a) First Order Motion Model+ Discriminator

RGB image $x \in \mathbb{R}^{256\times256\times3}$
ConvBlock down $ch \rightarrow 2ch$
ConvBlock down $2ch \rightarrow 4ch$
ConvBlock down $4h \rightarrow 8ch$
Conv2d $8ch \rightarrow 1$

(b) U-Net Discriminator

RGB image $x \in \mathbb{R}^{256\times256\times3}$
ConvBlock down $ch \rightarrow 2ch$
ConvBlock down $2ch \rightarrow 4ch$
ConvBlock down $4h \rightarrow 8ch$ *(see below)
ConvBlock up $8ch \rightarrow 4ch$
ConvBlock up $(4+4)ch \rightarrow 2ch$
ConvBlock up $(2+2)ch \rightarrow ch$
ConvBlock up $(ch+ch) \rightarrow ch$
Conv2d $ch \rightarrow 1$
Sigmoid
*Conv2d $8ch \rightarrow 1$

Table A1: The architectures of First Order Motion Model+ discriminator and our U-Net discriminator.

A.2 IMAGE GENERATION MODULE

The image generation module takes into account occluded parts in $\mathbf{S}$ that cannot be recovered by image warping and inpaints those parts based on an occlusion map $\hat{\mathcal{O}}_{\mathbf{S}\leftarrow\mathbf{D}} \in [0,1]^{H'\times W'}$. To obtain the occlusion map $\hat{\mathcal{O}}_{\mathbf{S}\leftarrow\mathbf{D}}$ from the sparse keypoint representation, a channel is added to the final layer of the dense motion network. Suppose $\boldsymbol{\xi} \in \mathbb{R}^{H'\times W'}$ is the feature map obtained from the down-sampling convolutional blocks. The transformed feature map $\boldsymbol{\xi}'$ is obtained as follows:

$$\boldsymbol{\xi}' = \hat{\mathcal{O}}_{\mathbf{S}\leftarrow\mathbf{D}} \odot f_w(\boldsymbol{\xi}, \hat{\mathcal{T}}_{\mathbf{S}\leftarrow\mathbf{D}}) \tag{6}$$

where f_w denotes the back-warping operation and $\odot$ denotes the Hadamard product. The subsequent layers of the image generation module decode the transformed feature map $\boldsymbol{\xi}'$ back to the target image.

B ARCHITECTURES AND TRAINING DETAILS

Architectures We followed Schonfeld et al. (2020) to adapt a U-Net architecture in the discriminator. A U-Net discriminator D^U consists of both an encoder D^U_{enc} and a decoder D^U_{dec}. The U-Net encoder D^U_{enc} progressively downsamples the input image to predict global realism like the discriminator of a vanilla GAN. The U-Net decoder D^U_{dec} progressively upsamples the feature map to predict the realism of individual pixels. It uses skip connections to feed information between matching resolutions of D^U_{enc} and D^U_{dec}, enhancing its capability in segmenting fine details. The U-Net discriminator learns both the global and the local differences between real and fake images. Table A1 summarizes the differences between our discriminator architecture and that of First Order Motion Model+[1], the concurrent work of First Order Motion Model (Siarohin et al., 2019) with a global discriminator. Our U-Net encoder uses the same architecture like that of the global discriminator of First Order Motion Model+. Our U-Net decoder is a symmetry of the U-Net encoder architecture, with skip connections feeding information between the matching resolutions of the two modules. Both discriminators use $ch = 64$ for channels. For the keypoint detector and the generator, we use the same architectures as those of the baseline First Order Motion Model.

Hyperparameters We followed Siarohin et al. (2019) to use the Adam (Kingma & Ba, 2014) optimizer for training with learning rate 2e-4 and batch size 20. For regularization with PriorityCut, we followed Schonfeld et al. (2020) to linearly increase the probability of augmentation from 0 to 0.5

[1] https://github.com/AliaksandrSiarohin/first-order-model

for the first n epochs. This gives the generator sufficient time to warm up and does not make the discriminator too strong in the beginning.

Training losses Following Siarohin et al. (2019) and Schonfeld et al. (2020), we used a combination of losses for both the discriminator and the generator. The loss of the U-Net discriminator D^U consists of the adversarial loss of the U-Net encoder $\mathcal{L}_{D^U_{enc}}$, the adversarial loss of the U-Net decoder $\mathcal{L}_{D^U_{dec}}$, and the consistency regularization loss $\mathcal{L}^{cons}_{D^U_{dec}}$ for the CutMix operation between the real and the fake images.

$$\mathcal{L}_{D^U} = \mathcal{L}_{D^U_{enc}} + \mathcal{L}_{D^U_{dec}} + \lambda \mathcal{L}^{cons}_{D^U_{dec}} \tag{7}$$

In particular, the consistency regularization loss consists of two terms. The first term is the output of the U-Net decoder D^U_{dec} on the CutMix image. The second term is the CutMix between the outputs of D^U_{dec} on real and fake images. The consistency regularization loss computes the L^2 norm between the first and the second terms.

$$\mathcal{L}^{cons}_{D^U_{dec}} = \left\| D^U_{dec}\left(\text{mix}(x, x', \min_k \hat{\mathcal{O}}_{fg})\right) - \text{mix}\left(D^U_{dec}(x), D^U_{dec}(x'), \min_k \hat{\mathcal{O}}_{fg}\right)\right\|^2 \tag{8}$$

The loss of the generator consists of the reconstruction loss $\mathcal{L}_{rec}$, the equivariance loss $\mathcal{L}_{equiv}$, the adversarial loss $\mathcal{L}_{adv}$, and the feature matching loss $\mathcal{L}_{feat}$. The published First Order Motion Model paper uses only the reconstruction loss $\mathcal{L}_{rec}$ and the equivariance loss $\mathcal{L}_{equiv}$ (Siarohin et al., 2019). We followed its concurrent work to include the adversarial loss $\mathcal{L}_{adv}$ and the feature matching loss $\mathcal{L}_{feat}$ for adversarial training.

$$\mathcal{L}_G = \mathcal{L}_{rec} + \mathcal{L}_{equiv} + \mathcal{L}_{adv} + \mathcal{L}_{feat} \tag{9}$$

The reconstruction loss $\mathcal{L}_{rec}$ is the multi-scale perceptual loss based on the activations of the pre-trained VGG-19 network (Simonyan & Zisserman, 2014) between real and fake images. The equivariance loss $\mathcal{L}_{equiv}$ encourages consistent keypoint predictions given known geometric transformations. It is based on the following equivariance constraint on local motion approximations:

$$\mathcal{T}_{\mathbf{X}\leftarrow\mathbf{R}} \equiv \mathcal{T}_{\mathbf{X}\leftarrow\mathbf{Y}} \circ \mathcal{T}_{\mathbf{Y}\leftarrow\mathbf{R}} \tag{10}$$

The adversarial loss $\mathcal{L}_{adv}$ is the sum of the feedback from the encoder D^U_{enc} and the decoder D^U_{dec} of the U-Net discriminator D^U. The feature matching loss $\mathcal{L}_{feat}$ matches the feature maps of each layer of the U-Net encoder $\mathcal{L}_{D^U_{enc}}$ between the driving and the generated images, similar to that of pix2pixHD (Wang et al., 2018).

C Experimental Setup Details

Datasets We followed the preprocessing protocols of Siarohin et al. (2019) to obtain high-quality videos on the following three datasets. At the time of data collection, some videos are no longer available on YouTube. We report the number of videos used in our experiments.

- The VoxCeleb dataset (Nagrani et al., 2017) is a face dataset with 22,496 videos from YouTube. We tracked the face until it is too far from its initial position and cropped the frames using the smallest crop containing all the bounding boxes. Then, we removed videos of resolution lower than 256×256 and resized the remaining videos to 256×256. After preprocessing, we obtained 18,398 videos for training and 512 videos for testing.
- The BAIR robot pushing dataset (Ebert et al., 2017) contains videos of a Sawyer robotic arm pushing objects over a table. It contains 42,880 videos for training and 128 videos for testing.
- The Tai-Chi-HD dataset (Siarohin et al., 2019) contains 280 tai-chi videos from YouTube. We used similar preprocessing steps as VoxCeleb to split the videos into short clips and resized all high-quality videos to 256×256. After preprocessing, we obtained 2,994 video chunks for training and 285 video chunks for testing.

Unlike First Order Motion Model paper (Siarohin et al., 2019), we did not experiment on the UvA-NEMO dataset (Dibeklioğlu et al., 2012), since the VoxCeleb dataset is a more difficult one to learn. The VoxCeleb dataset contains videos of different pose angles while the UvA-NEMO dataset contains only frontal faces. Also, the UvA-NEMO dataset has a simple uniform dark background and the facial features are the only moving parts with subtle movements.

Metrics We provide additional details on the metrics and the third-party tools used for computation.

- $\mathcal{L}_1$. This measures the pixel-wise differences between the generated and the ground truth videos.
- *Peak Signal-to-Noise Ratio (PSNR)*. This measures the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. It evaluates how good the model is at reconstructing the low-level details of the ground truth videos.
- *Structural Similarity (SSIM)*. This compares the low-level structures between the generated and the ground truth videos. It evaluates also how good the model is at reconstructing the low-level details.
- *Masked PSNR and SSIM (M-PSNR, M-SSIM)*. We used a variety of masks to evaluate the masked PSNR and SSIM on both the foreground and the background. The salient masks in Tables 1 and 2 cover the most noticeable parts of the images. The top-k masks in Tables A2, A3 and A4 correspond to the parts of the images that are the most difficult to inpaint.

  To compute a salient mask, we first used DeepLabv3+ (Chen et al., 2018), a semantic image segmentation library, to generate a binary mask. Then, we used an automatic trimap generator (Nugraha, 2018) to generate a trimap from the binary mask. We used the image matting library F, B, α Matting (Forte & Pitié, 2020) to generate a final alpha mask for the foreground. The background mask is the inversion of the foreground mask.

  We used the same methodology as Section 3.2 to obtain the masks based on the top-k percent occluded pixels. The top-k masks are derived from the occlusion masks predicted by the *baseline* model (FOMM). Similar to salient masks, we used the top-k masks to evaluate the foreground and their inversions to evaluate the background.
- *Average Keypoint Distance (AKD)*. This measures the average keypoint distance between the generated and the ground truth videos. Following Siarohin et al. (2019), we used a face alignment library (Bulat & Tzimiropoulos, 2017) to detect face keypoints for the VoxCeleb dataset and a human pose estimation library (Cao et al., 2017) to detect pose keypoints for the Tai-Chi-HD dataset.
- *Missing Keypoint Rate (MKR)*. This measures the percentage of keypoints detected in the ground truth videos but not in the generated ones. This evaluates the appearance quality of the generated videos. Following Siarohin et al. (2019), we used the binary label returned by the human pose estimation library (Cao et al., 2017) to count if a keypoint is detected or not for the Tai-Chi-HD dataset.
- *Average Euclidean Distance (AED)*. This measures the distance of the feature embedding between the generated and the ground truth videos. Following Siarohin et al. (2019), we used OpenFace (Baltrusaitis et al., 2018) to extract the face identity embedding for the VoxCeleb dataset and a person re-id library (Hermans et al., 2017) to extract the person re-identification embedding for the Tai-Chi-HD dataset.

D Additional Evaluations

Quantitative comparison We additionally evaluated masked PSNR and SSIM on state-of-the-art approaches based on the top-k percent occluded pixels. In Tables A2, A3 and A4, the columns *top-k* represent the masks of the heaviest k percent occluded pixels. The columns $\neg$ *top-k* are the inversions of the top-k masks. We evaluated both masks to check if the models compromise the quality of certain parts of the image to achieve the desired performance. For both metrics, the larger the values, the better the results. The bold texts represent the best results. The red and green texts represent performance loss and gain compared to the baseline model (FOMM), respectively.

We evaluated the masked versions on the VoxCeleb, BAIR, and Tai-Chi-HD datasets. Table A2 shows the results for VoxCeleb. For both PSNR and SSIM, PriorityCut outperforms state-of-the-art approaches in different thresholds k. Note that adversarial training alone does not guarantee performance gains. For PSNR, the adversarial model (FOMM+) compromises the quality of the hardest parts to inpaint (top-k masks) and pursues the easy targets ($\neg$ top-k masks). For SSIM, adversarial training performs worse in every setting. Table A3 shows the results for BAIR. For both PSNR and SSIM, PriorityCut consistently outperforms state-of-the-art approaches. Table A4 shows the results for Tai-Chi-HD. For PSNR, PriorityCut outperforms state-of-the-art approaches in all settings. For SSIM, PriorityCut is on par with the adversarial model.

k	10%		20%		30%		40%		50%	
	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k
X2Face	28.42±0.02	19.94±0.02	25.41±0.02	20.74±0.02	23.72±0.02	21.59±0.02	22.57±0.02	22.51±0.02	21.70±0.02	23.55±0.02
Monkey-Net	30.81±0.02	23.50±0.02	28.00±0.02	24.44±0.02	26.45±0.02	25.41±0.02	25.41±0.02	26.44±0.02	24.65±0.02	27.57±0.02
FOMM	32.99±0.02	25.24±0.02	30.07±0.02	26.14±0.02	28.46±0.02	27.09±0.02	27.37±0.02	28.11±0.02	26.55±0.02	29.24±0.02
FOMM+	32.91±0.02	25.24±0.02	30.00±0.02	26.16±0.02	28.39±0.02	27.12±0.02	27.30±0.02	28.15±0.02	26.49±0.02	29.30±0.02
Ours	**33.14**±0.02	**25.42**±0.02	**30.21**±0.02	**26.34**±0.02	**28.60**±0.02	**27.29**±0.02	**27.51**±0.02	**28.32**±0.02	**26.70**±0.02	**29.46**±0.02

(a) Masked PSNR on top-k percent occluded pixels.

k	10%		20%		30%		40%		50%	
	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k
X2Face	0.9519±9e-5	0.6792±5e-4	0.9108±1e-4	0.7262±5e-4	0.8721±2e-4	0.7702±4e-4	0.8344±2e-4	0.8114±4e-4	0.7977±3e-4	0.8499±3e-4
Monkey-Net	0.9640±7e-5	0.7755±4e-4	0.9333±1e-4	0.8134±4e-4	0.9041±2e-4	0.8473±3e-4	0.8757±2e-4	0.8779±3e-4	0.8483±3e-4	0.9053±2e-4
FOMM	0.9736±6e-5	0.8270±4e-4	0.9497±1e-4	0.8573±3e-4	0.9264±2e-4	0.8842±3e-4	0.9034±2e-4	0.9085±2e-4	0.8813±3e-4	0.9301±2e-4
FOMM+	0.9734±6e-5	0.8259±4e-4	0.9493±1e-4	0.8563±3e-4	0.9260±2e-4	0.8834±3e-4	0.9028±2e-4	0.9079±2e-4	0.8805±3e-4	0.9297±2e-4
Ours	**0.9741**±6e-5	**0.8287**±4e-4	**0.9505**±1e-4	**0.8587**±1e-4	**0.9275**±2e-4	**0.8854**±3e-4	**0.9048**±2e-4	**0.9094**±2e-4	**0.8829**±2e-4	**0.9308**±2e-4

(b) Masked SSIM on top-k percent occluded pixels.

Table A2: Comparison with state-of-the-art for approaches for video reconstruction on VoxCeleb.

k	10%		20%		30%		40%		50%	
	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k
X2Face	24.88±0.1	24.86±0.1	23.10±0.1	27.42±0.2	22.39±0.1	29.53±0.2	22.01±0.1	31.42±0.2	21.78±0.1	33.23±0.2
Monkey-Net	27.11±0.1	26.21±0.1	25.09±0.1	28.69±0.2	24.29±0.1	30.80±0.2	23.86±0.1	32.74±0.1	23.59±0.1	34.70±0.1
FOMM	29.94±0.1	27.24±0.1	27.51±0.1	29.40±0.1	26.48±0.1	31.38±0.1	25.91±0.1	33.27±0.1	25.53±0.1	35.21±0.1
Ours	**30.71**±0.1	**27.61**±0.1	**28.19**±0.1	**29.73**±0.1	**27.11**±0.1	**31.67**±0.1	**26.49**±0.1	**33.53**±0.1	**26.09**±0.1	**35.42**±0.1

(a) Masked PSNR on top-k percent occluded pixels.

k	10%		20%		30%		40%		50%	
	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k
X2Face	0.9409±4e-4	0.8885±2e-3	0.9097±9e-4	0.9228±1e-3	0.8906±1e-3	0.9429±1e-3	0.8772±1e-3	0.9564±8e-4	0.8670±2e-3	0.9661±7e-4
Monkey-Net	0.9608±4e-4	0.9112±1e-3	0.9355±8e-4	0.9393±1e-3	0.9188±1e-3	0.9565±8e-4	0.9068±1e-3	0.9681±6e-4	0.8977±1e-3	0.9765±4e-4
FOMM	0.9723±4e-4	0.9218±1e-3	0.9512±7e-4	0.9449±9e-4	0.9365±9e-4	0.9601±7e-4	0.9255±1e-3	0.9707±5e-4	0.9169±5e-4	0.9787±4e-4
Ours	**0.9750**±3e-4	**0.9244**±1e-3	**0.9551**±6e-4	**0.9463**±9e-4	**0.9409**±9e-4	**0.9609**±7e-4	**0.9304**±1e-3	**0.9714**±5e-4	**0.9221**±1e-3	**0.9791**±4e-4

(b) Masked SSIM on top-k percent occluded pixels.

Table A3: Comparison with state-of-the-art for approaches for video reconstruction on BAIR.

Qualitative comparison We performed additional qualitative comparisons between state-of-the-art approaches on each dataset.

Figures 1 and 2 show the results for VoxCeleb. X2Face shows severe warping artifacts and face distortions. Monkey-Net does not follow the pose angles properly and shows warping artifacts around locations of large changes in motion (hair or area around the edges of the faces). FOMM has difficulty distinguishing between foreground and background texture around those locations. FOMM+ amplifies the foreground or background artifacts at those locations. In contrast, PriorityCut shows a clear distinction between foreground and background and does not inpaint confusing texture.

Figure 3 shows the results for BAIR. X2Face shows trivial warping artifacts and broken texture on the robot arms. Monkey-Net shows warping artifacts around the robot arms. FOMM prefers to erase the background objects when it fails to recover the artifacts and produces blurry texture on the robot arms when they are close to the edge of the frame. In contrast, PriorityCut preserves better texture of both the robot arms and the background objects.

k	10%		20%		30%		40%		50%	
	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k
X2Face	27.52±0.03	19.02±0.02	25.05±0.03	19.64±0.02	23.74±0.03	20.16±0.02	22.83±0.03	20.66±0.02	22.21±0.03	21.11±0.02
Monkey-Net	28.11±0.03	19.81±0.03	25.68±0.03	20.43±0.03	24.35±0.03	20.97±0.03	23.43±0.03	21.50±0.03	22.79±0.03	21.98±0.03
FOMM	31.50±0.04	22.01±0.03	28.62±0.03	22.62±0.03	27.05±0.03	23.20±0.03	25.97±0.03	23.78±0.03	25.21±0.03	24.34±0.03
FOMM+	31.61±0.04	22.08±0.03	28.68±0.03	22.70±0.03	27.09±0.03	23.29±0.03	26.00±0.03	23.88±0.03	25.24±0.03	24.45±0.03
Ours	**31.76**±0.04	**22.26**±0.03	**28.86**±0.03	**22.88**±0.03	**27.28**±0.03	**23.47**±0.03	**26.18**±0.03	**24.06**±0.03	**25.42**±0.03	**24.63**±0.03

(a) Masked PSNR on top-k percent occluded pixels.

k	10%		20%		30%		40%		50%	
	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k	top-k	$\neg$ top-k
X2Face	0.9477±2e-4	0.6439±1e-3	0.9069±3e-4	0.6928±1e-3	0.8701±5e-4	0.7351±1e-3	0.8348±6e-4	0.7736±9e-4	0.8016±8e-4	0.8081±7e-4
Monkey-Net	0.9505±2e-4	0.6613±1e-3	0.9116±4e-4	0.7077±1e-3	0.8761±5e-4	0.7478±1e-3	0.8422±7e-4	0.7846±9e-4	0.8102±9e-4	0.8177±7e-4
FOMM	0.9626±2e-4	0.7034±1e-3	0.9301±3e-4	0.7441±1e-3	0.8992±5e-4	0.7803±1e-3	0.8687±7e-4	0.8140±8e-4	0.8393±8e-4	0.8447±7e-4
FOMM+	**0.9633**±2e-4	**0.7055**±1e-3	**0.9310**±3e-4	**0.7459**±1e-3	**0.9002**±5e-4	**0.7822**±1e-3	**0.8697**±7e-4	0.8159±8e-4	**0.8404**±8e-4	0.8465±7e-4
Ours	0.9631±2e-4	0.7048±1e-3	0.9307±3e-4	0.7456±1e-3	0.8997±5e-4	0.7820±1e-3	0.8690±7e-4	**0.8160**±8e-4	0.8393±8e-4	**0.8470**±7e-4

(b) Masked SSIM on top-k percent occluded pixels.

Table A4: Comparison with state-of-the-art for approaches for video reconstruction on Tai-Chi-HD.

Figure 4 shows the results for Tai-Chi-HD. X2Face shows slight to severe warping artifacts, depending on the motion. MonkeyNet shows trivial warping artifacts in the background and around the hands. FOMM produces visible warping artifacts on the faces and FOMM+ amplifies the artifacts. In contrast, PriorityCut preserves better identity without trivial warping artifacts on the faces.

REFERENCES

Tadas Baltrusaitis, Amir Zadeh, Yao Chong Lim, and Louis-Philippe Morency. Openface 2.0: Facial behavior analysis toolkit. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pp. 59–66. IEEE, 2018.

Adrian Bulat and Georgios Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem? (and a dataset of 230,000 3d facial landmarks). In *International Conference on Computer Vision*, 2017.

Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7291–7299, 2017.

Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 801–818, 2018.

Hamdi Dibeklioğlu, Albert Ali Salah, and Theo Gevers. Are you really smiling at me? spontaneous versus posed enjoyment smiles. In *European Conference on Computer Vision*, pp. 525–538. Springer, 2012.

Frederik Ebert, Chelsea Finn, Alex X Lee, and Sergey Levine. Self-supervised visual planning with temporal skip connections. *arXiv preprint arXiv:1710.05268*, 2017.

Marco Forte and François Pitié. f, b, alpha matting. *arXiv preprint arXiv:2003.07711*, 2020.

Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

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

Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. Voxceleb: a large-scale speaker identification dataset. *arXiv preprint arXiv:1706.08612*, 2017.

Leo Nugraha. Automatic trimap generator, 2018. URL https://github.com/lnugraha/trimap_generator.

Edgar Schonfeld, Bernt Schiele, and Anna Khoreva. A u-net based discriminator for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8207–8216, 2020.

Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. In *Advances in Neural Information Processing Systems*, pp. 7135–7145, 2019.

Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Figure 1: Qualitative comparison of state-of-the-art approaches for image animation on VoxCeleb.

Figure 2: Qualitative comparison of state-of-the-art approaches for image animation on VoxCeleb.

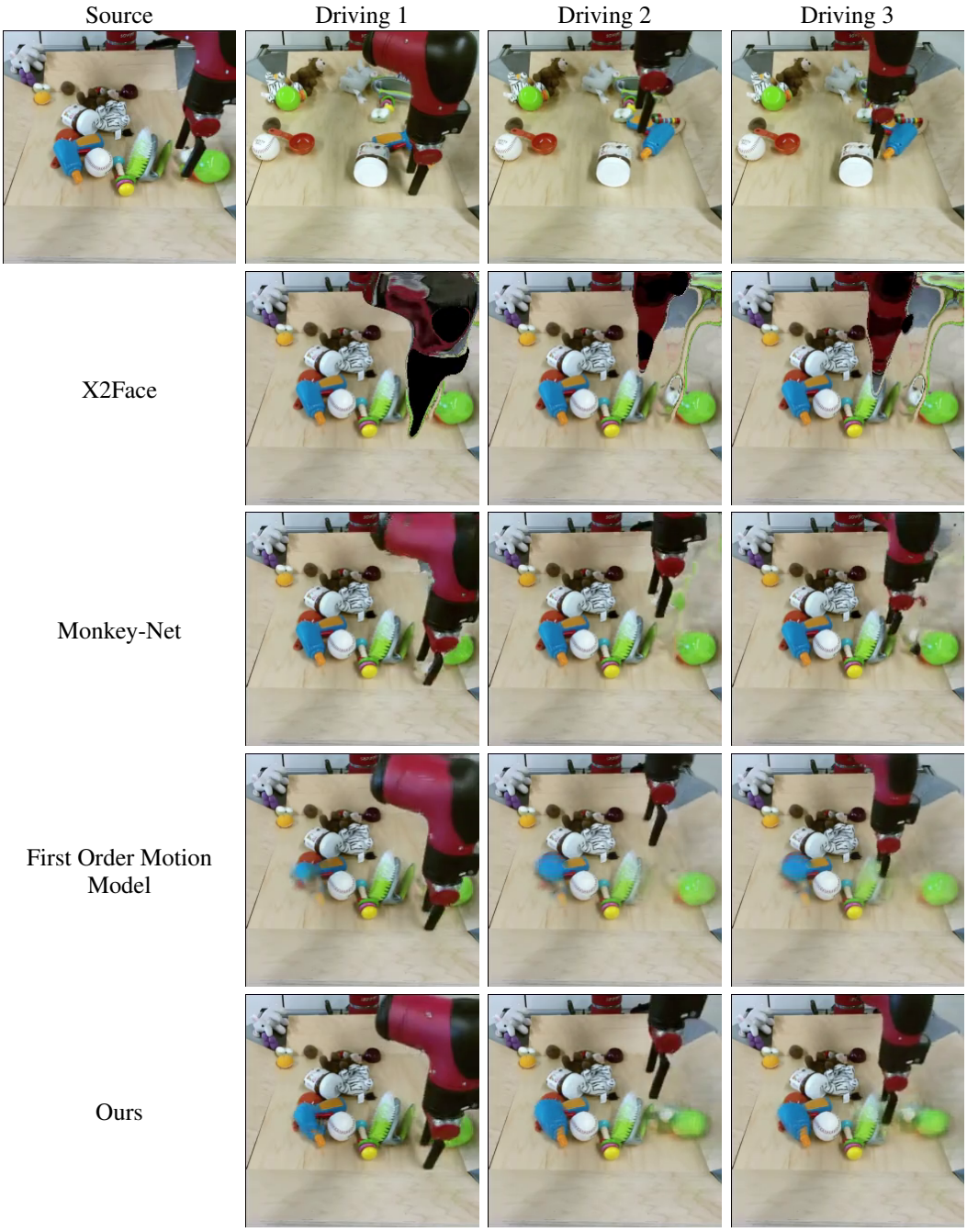

Figure 3: Qualitative comparison of state-of-the-art approaches for image animation on BAIR.

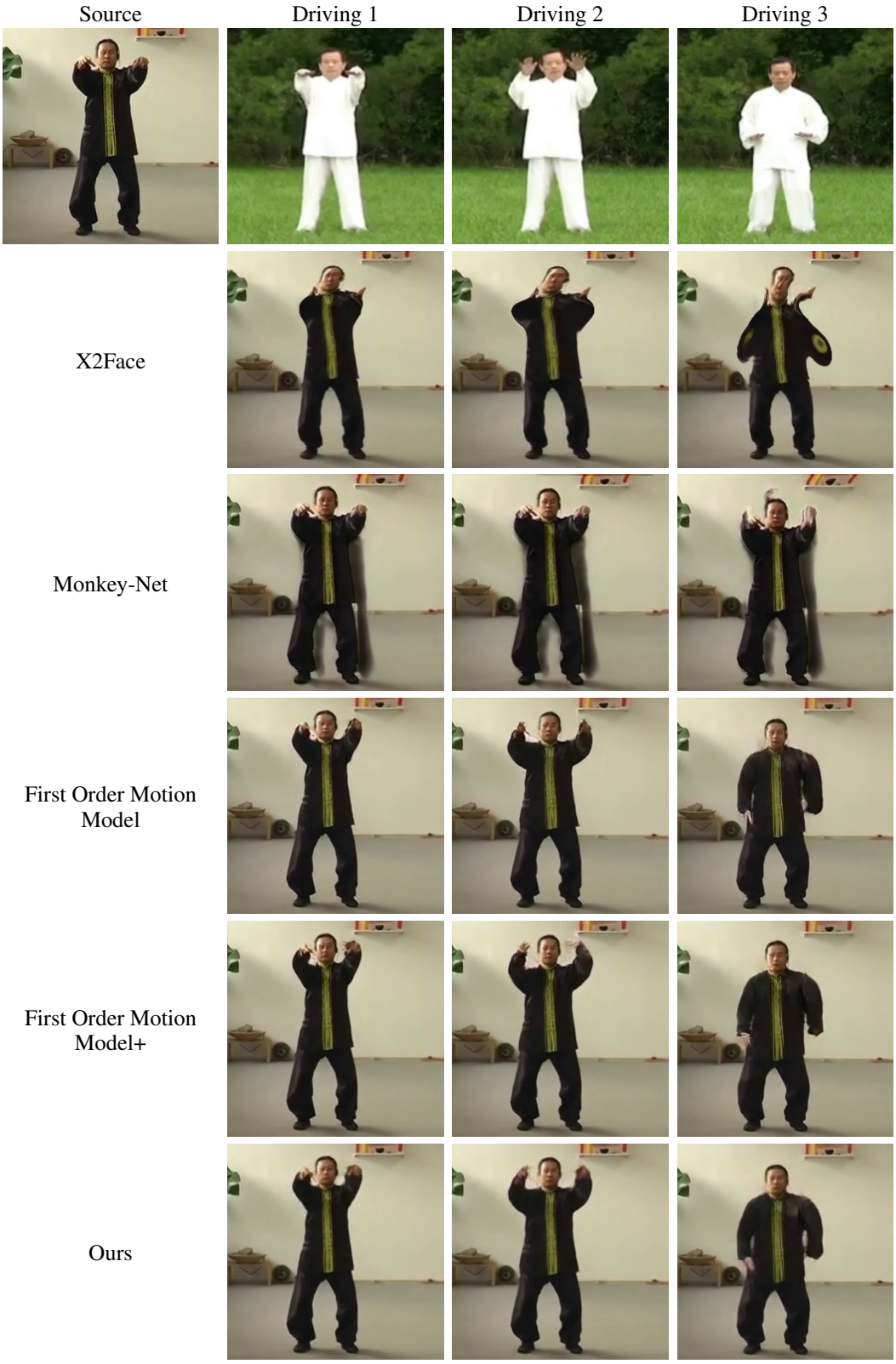

Figure 4: Qualitative comparison of state-of-the-art approaches for image animation on Tai-Chi-HD.