Supplementary Material:
Cross-Identity Motion Transfer for Arbitrary Objects through Pose-Attentive Video Reassembling

Anonymous ECCV submission

Paper ID 4585

1 Ablation study on the number of keypoints

Our method extracts K keypoints internally to describe object's pose in images. To show the effect of K on the performance, we conducted an ablation study on the number of keypoints on VoxCeleb2 dataset [1]. In Fig. 1, the top row shows FID and AKD scores in the setting of self-identity motion transfer, while the bottom shows FID and AED scores in the setting of cross-identity transfer. According to FID scores, the overall quality of generated images is similar regardless of K. However, we observe a considerable improvement in AKD and AED when using 64 keypoints, implying that the larger number of keypoints is effective in transferring motion more accurately.

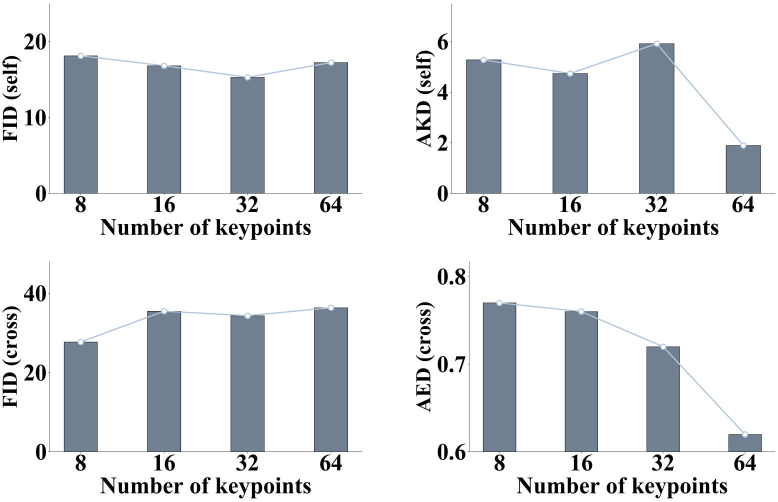


Fig. 1. Ablation study on the number of keypoints. The top and the bottom row show quantitative results on self-identity and cross-identity settings, respectively.

2 Implementation details

We implement our network using PyTorch [3]. We train our network using Adam optimizer with $\beta_1 = 0.5$, $\beta_2 = 0.99$. We initialize the learning rate to 0.0002 for 100,000 iterations, and we linearly decay the learning rate to 0 over the rest of iterations. We use 4 GPUs of Geforce RTX 2080 TI for training, and the training took about 4 days. Our network set the number of keypoints K as 64 for Thai-chi-HD [5] and VoxCeleb2 [1] datasets, and K is 30 for BAIR robot arm dataset [2].

The detailed network structure is presented in Table 1. Here, Conv2d(K, S, P) indicates 2D convolution with the kernel size of K, the stride of S, and the padding of P. In Interpolate(S_x, S_y), S_x and S_y represent the scale factor of interpolation. BN indicates batch normalization and we use LeakyReLUs with slope of 0.1 as the activation function.

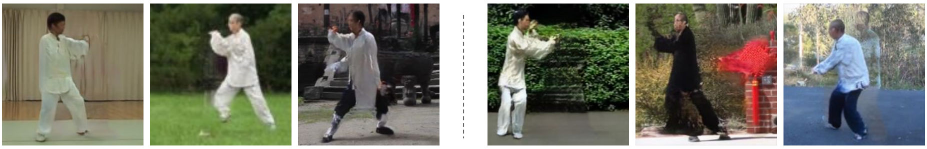
Table 1. Network details. (* indicates parallel layers)

Layer	Output size
Encoder	
(*input*)	$224 \times 224 \times 3$
Conv2d(7,1,3), BN, LeakyReLU Conv2d(3,1,1), BN, LeakyReLU	$224 \times 224 \times 32$
Conv2d(4,2,1), BN, LeakyReLU Conv2d(3,1,1), BN, LeakyReLU	$112 \times 112 \times 64$
Conv2d(4,2,1), BN, LeakyReLU Conv2d(3,1,1), BN, LeakyReLU	$56 \times 56 \times 128$
Conv2d(4,2,1), BN, LeakyReLU Conv2d(3,1,1), BN, LeakyReLU	$28 \times 28 \times 256$
Conv2d(4,2,1), BN, LeakyReLU Conv2d(3,1,1), BN, LeakyReLU	$14 \times 14 \times 512$

Layer	Output size
Decoder	
(*input*)	$14 \times 14 \times 256$
Interpolate(2,2) Conv2d(3,1,1), BN, LeakyReLU Conv2d(3,1,1), BN, LeakyReLU	$28 \times 28 \times 256$
Interpolate(2,2) Conv2d(3,1,1), BN, LeakyReLU Conv2d(3,1,1), BN, LeakyReLU	$56 \times 56 \times 128$
Interpolate(2,2) Conv2d(3,1,1), BN, LeakyReLU Conv2d(3,1,1), BN, LeakyReLU	$112 \times 112 \times 64$
Interpolate(2,2) Conv2d(3,1,1), BN, LeakyReLU Conv2d(3,1,1), BN, LeakyReLU	$224 \times 224 \times 32$
* Conv2d(7,1,3), Tanh	$224 \times 224 \times 3$
* Conv2d(7,1,3), Sigmoid	$224 \times 224 \times 1$

3 Failure Cases

Fig. 2 depicts some failure cases on Thai-Chi-HD dataset [5]. As shown in Fig. 2 (a), our method sometimes fails to capture small regions such as hands or feet. Background inpainting is another challenge for motion transfer as shown in Fig. 2 (b). Since our training is mainly focused on minimizing reconstruction error, newly synthesized background may not be perfect.



(a) hands and feet

(b) background inpainting

Fig. 2. Failure cases on Thai-Chi-HD dataset.

4 Additional results

In figures 3,4, and 5, we show more results on the VoxCeleb2, Thai-Chi-HD, and BAIR datasets. We compare our method with X2Face [6], Monkey-net [4], and First-order methods [5] using a single source image. More results using multiple source images are presented in Fig. 6.

References

1. Chung, J.S., Nagrani, A., Zisserman, A.: Voxceleb2: Deep speaker recognition. Proc. Interspeech 2018 pp. 1086–1090 (2018)
2. Ebert, F., Finn, C., Lee, A.X., Levine, S.: Self-supervised visual planning with temporal skip connections. arXiv preprint arXiv:1710.05268 (2017)
3. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., Devito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017)
4. Siarohin, A., Lathuilière, S., Tulyakov, S., Ricci, E., Sebe, N.: Animating arbitrary objects via deep motion transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2377–2386 (2019)
5. Siarohin, A., Lathuillere, S., Tulyakov, S., Ricci, E., Sebe, N.: First order motion model for image animation. In: Advances in Neural Information Processing Systems 32, pp. 7135–7145 (2019)
6. Wiles, O., Sophia Koepke, A., Zisserman, A.: X2face: A network for controlling face generation using images, audio, and pose codes. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 670–686 (2018)

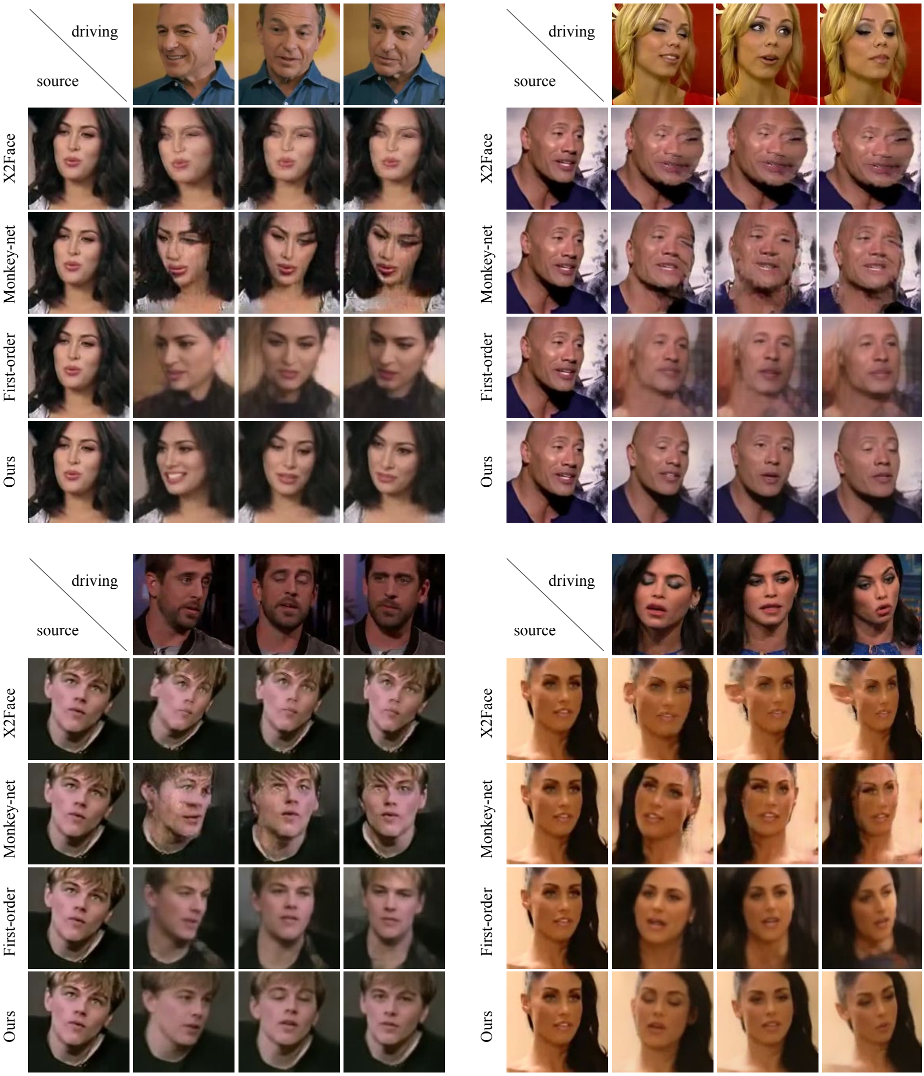


Fig. 3. VoxCeleb2 results of motion transfer using a single frame.

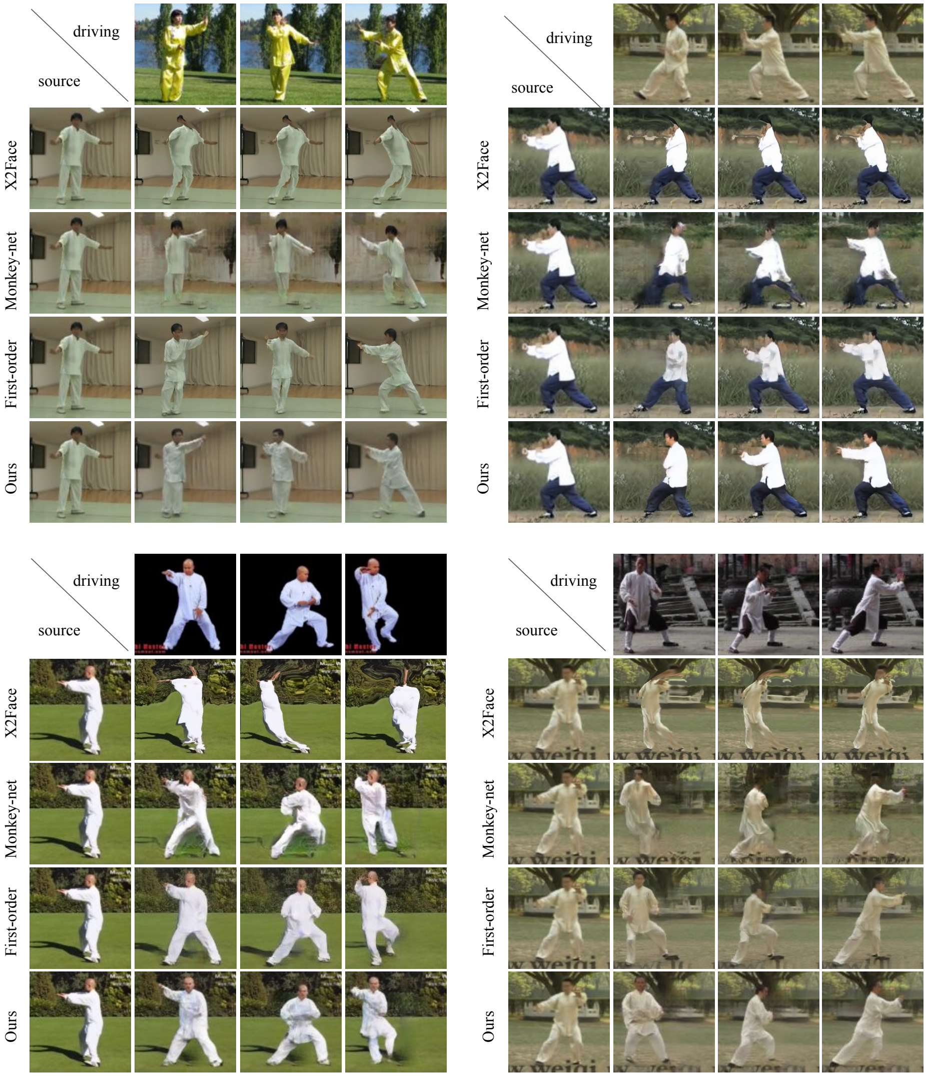


Fig. 4. Thai-Chi-HD results of motion transfer using a single frame.

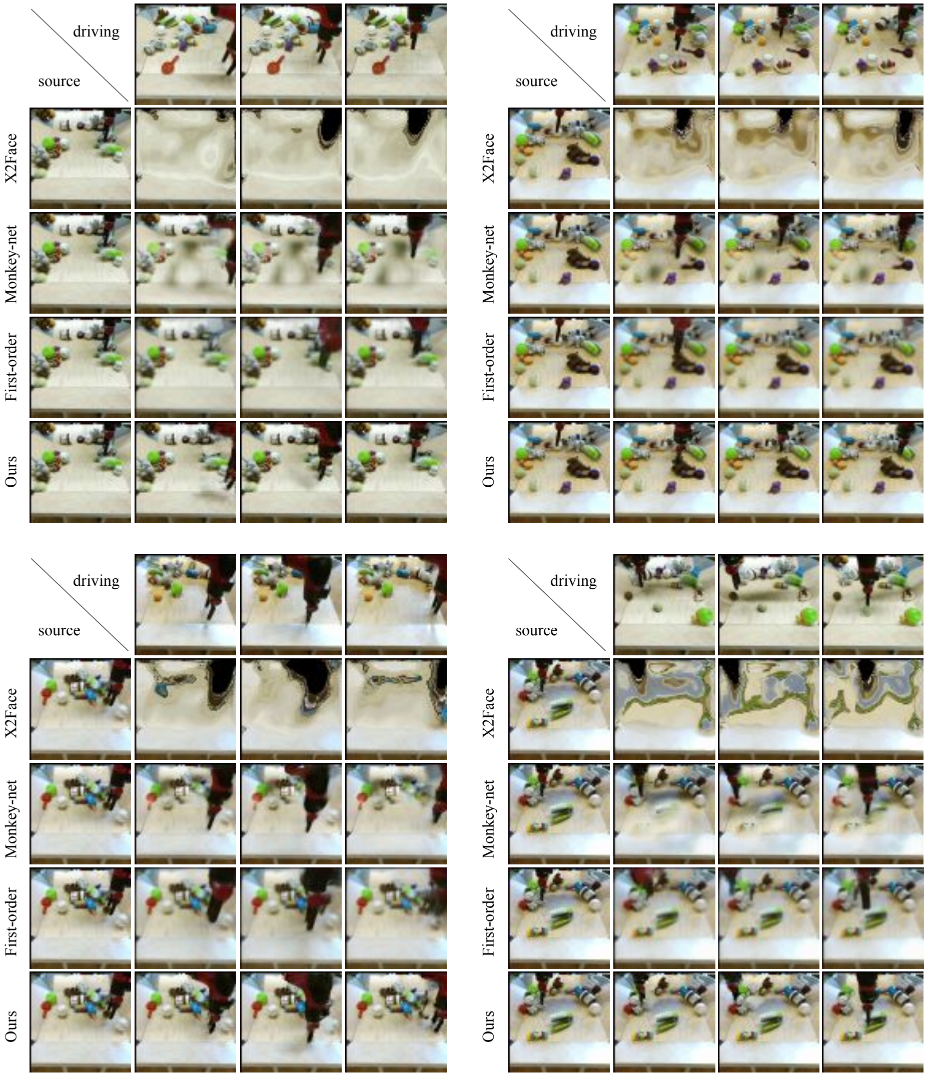


Fig. 5. BAIR results of motion transfer using a single frame.

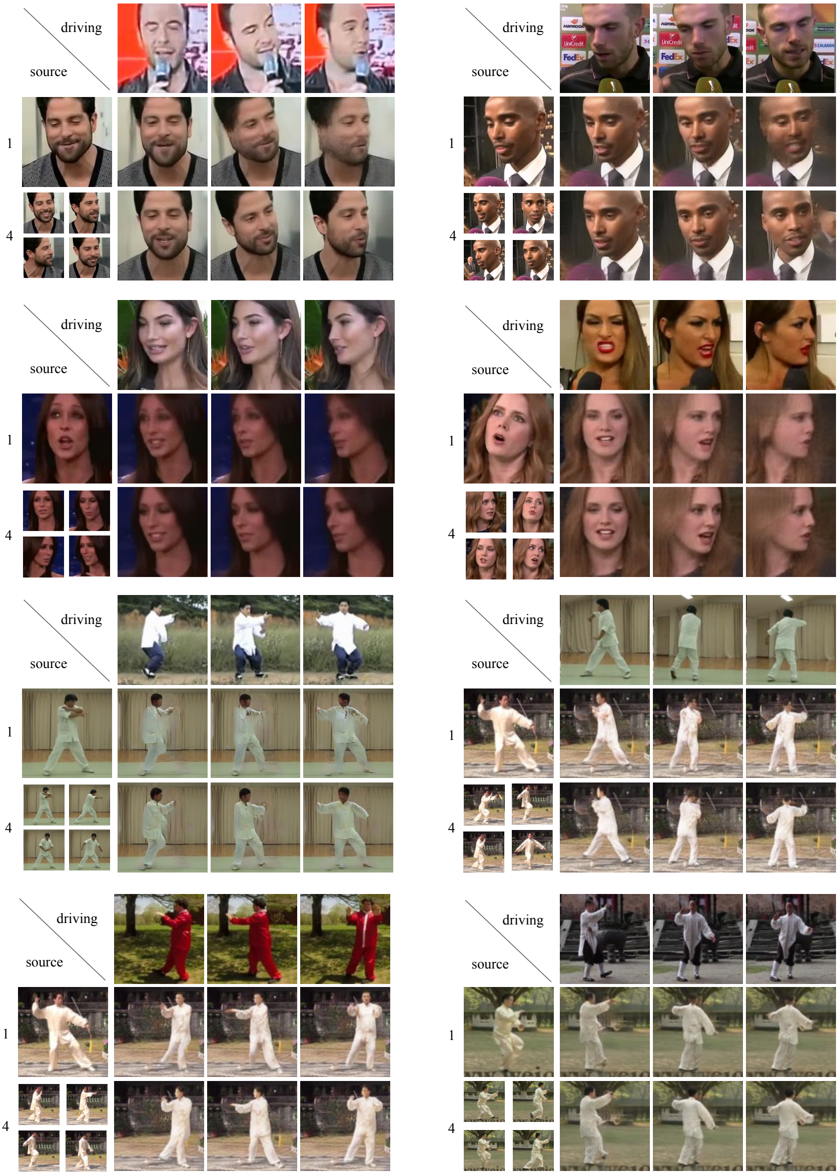


Fig. 6. VoxCeleb2 and Thai-Chi-HD results using multiple frames.