First Order Motion Model for Image Animation

NeurIPS'19

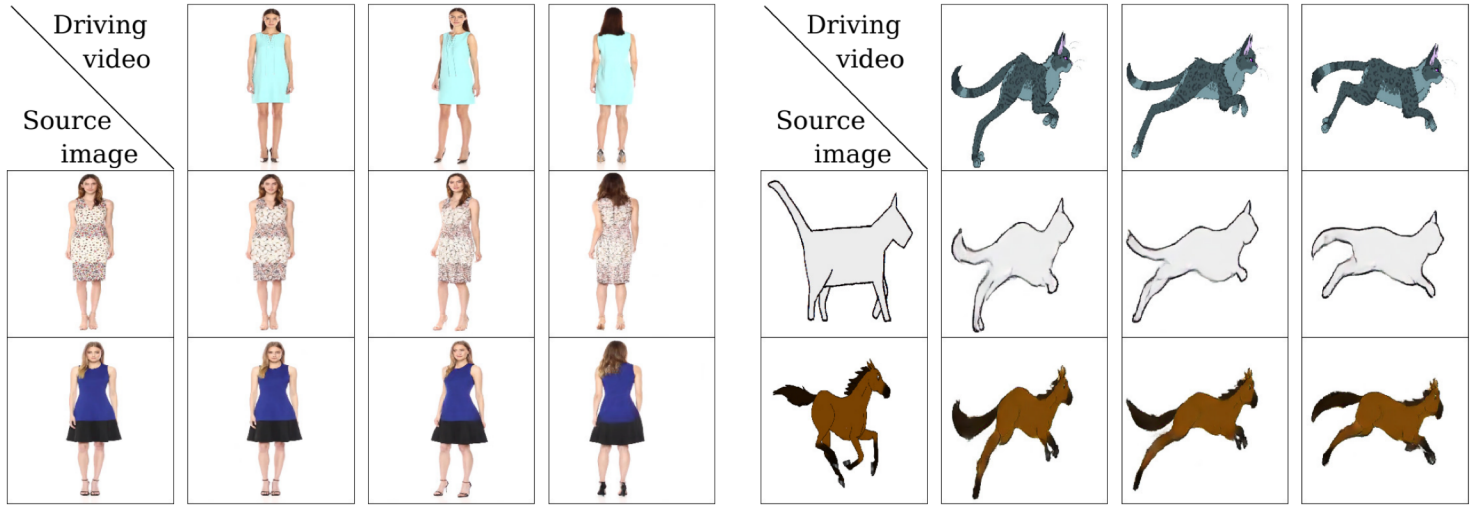
2021.02.08

Presented by Junsoo Lee

DAVIAN

Image Animation?

- Object (Image) Animation / Video Re-targeting:
  - Image animation consists of generating a video sequence so that an object in a source image is animated according to the motion of a driving video.
  - Image animation from a driving video can be interpreted as the problem of *transferring motion information* from one object to another.



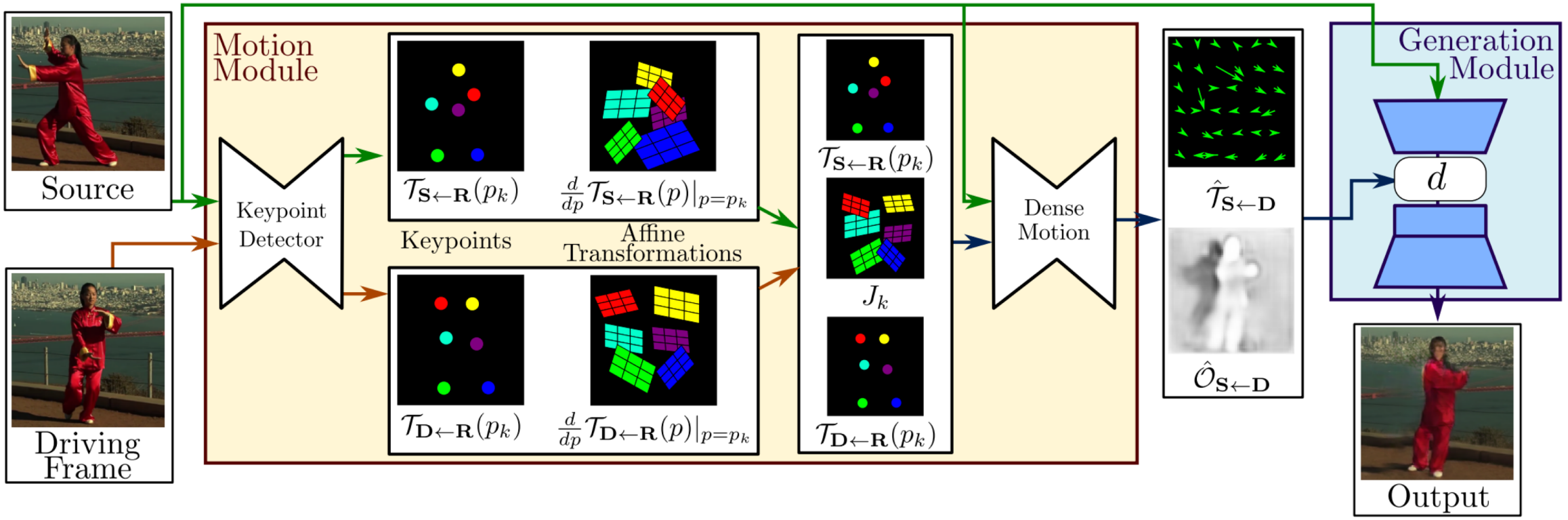
Keywords

- Self-learned keypoints
- Self-learned local affine transformations
  - To model complex motions
- Occlusion-aware generator
- Extended equivariance loss
  - To improve the estimation of local affine transformations

Method

- Since direct supervision is not available, we follow a self-supervised strategy.
- For training, our model is trained to reconstruct the training videos by combining a single frame and a learned motion representation in the video.
- At the test time, we apply our model to pairs composed of the source image and of each frame of the driving video and perform image animation of the source object.

Method: overall



Method: overall

- Motion estimation module:
  - Aims to predict a dense motion field $\hat{\mathcal{T}}_{S\leftarrow D}$ from D to S.
    - The dense motion field is used to align feature maps computed from S with the object pose in D.
  - We assume there exists an abstract reference frame R.
    - $\hat{\mathcal{T}}_{S\leftarrow R}$, $\hat{\mathcal{T}}_{D\leftarrow R}$.
  - The dense motion network outputs an occlusion mask $\hat{\mathcal{O}}_{S\leftarrow D}$.
- Image generartion module:
  - Warps the source image according to $\hat{\mathcal{T}}_{S\leftarrow D}$ and inpaints image parts that are occluded in the source image .

Method: motion estimation module / local affine transformation

- The motion estimation module predicts the backward optical flow $\mathcal{T}_{S\leftarrow D}$ from a driving frame D to the source frame S.
- Estimating $\mathcal{T}_{S\leftarrow D}$ consists in estimating $\mathcal{T}_{S\leftarrow R}$ and $\mathcal{T}_{R\leftarrow D}$.
- Given a frame X, we consider its first order Taylor expansion in K keypoints p_1, p_2, ..., p_K.

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

Method: motion estimation module / local affine transformation

- In order to estimate $\mathcal{T}_{X\leftarrow R}^{-1} = \mathcal{T}_{R\leftarrow X}$, we assume that $\mathcal{T}_{X\leftarrow R}$ is locally bijective in the neighbourhood of each keypoint.

$$\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{D}} = \mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}} \circ \mathcal{T}_{\mathbf{R}\leftarrow\mathbf{D}} = \mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}} \circ \mathcal{T}_{\mathbf{D}\leftarrow\mathbf{R}}^{-1}, \tag{3}$$

After computing again the first order Taylor expansion of Eq. (3) (see *Sup. Mat.*), we obtain:

$$\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{D}}(z) \approx \mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}}(p_k) + J_k(z - \mathcal{T}_{\mathbf{D}\leftarrow\mathbf{R}}(p_k)) \tag{4}$$

with:

$$J_k = \left(\frac{d}{dp}\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}}(p)\Big|_{p=p_k}\right)\left(\frac{d}{dp}\mathcal{T}_{\mathbf{D}\leftarrow\mathbf{R}}(p)\Big|_{p=p_k}\right)^{-1} \tag{5}$$

- In practice, $\mathcal{T}_{S\leftarrow R}$, $\mathcal{T}_{D\leftarrow R}$ are predicted by the keypoint predictor.

Method: motion estimation module / combining local motion

- We employ to estimate the dense motion field $\hat{\mathcal{T}}_{S\leftarrow D}$ from the set of Taylor approximations of $\mathcal{T}_{S\leftarrow D}(z)$ in the keypoints and the original source frame S.

For each keypoint p_k we additionally compute heatmaps $\mathbf{H}_k$ indicating to the dense motion network where each transformation happens. Each $\mathbf{H}_k(z)$ is implemented as the difference of two heatmaps centered in $\mathcal{T}_{\mathbf{D}\leftarrow\mathbf{R}}(p_k)$ and $\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}}(p_k)$:

$$\mathbf{H}_k(z) = exp\left(\frac{(\mathcal{T}_{\mathbf{D}\leftarrow\mathbf{R}}(p_k) - z)^2}{\sigma}\right) - exp\left(\frac{(\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}}(p_k) - z)^2}{\sigma}\right). \tag{6}$$

In all our experiments, we employ $\sigma = 0.01$ following Jakab *et al.* [18].

Method: motion estimation module / combining local motion

The heatmaps $\mathbf{H}_k$ and the transformed images $\mathbf{S}^0, \ldots \mathbf{S}^K$ are concatenated and processed by a U-Net [26]. $\hat{\mathcal{T}}_{\mathbf{S}\leftarrow\mathbf{D}}$ is estimated using a part-based model inspired by Monkey-Net [29]. We assume that an object is composed of K rigid parts and that each part is moved according to Eq. (4). Therefore we estimate K+1 masks $\mathbf{M}_k, k = 0, \ldots K$ that indicate where each local transformation holds. The final dense motion prediction $\hat{\mathcal{T}}_{\mathbf{S}\leftarrow\mathbf{D}}(z)$ is given by:

$$\hat{\mathcal{T}}_{\mathbf{S}\leftarrow\mathbf{D}}(z) = \mathbf{M}_0 z + \sum_{k=1}^{K} \mathbf{M}_k \left(\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}}(p_k) + J_k(z - \mathcal{T}_{\mathbf{D}\leftarrow\mathbf{R}}(p_k))\right) \tag{7}$$

Note that, the term $\mathbf{M}_0 z$ is considered in order to model non-moving parts such as background.

$$\mathcal{T}_{\mathbf{S}\leftarrow\mathbf{D}}(z) \approx \mathcal{T}_{\mathbf{S}\leftarrow\mathbf{R}}(p_k) + J_k(z - \mathcal{T}_{\mathbf{D}\leftarrow\mathbf{R}}(p_k)) \tag{4}$$

Method: Occlusion-aware image generation

- The source image S may be not aligned with the image to be generated video in the pixel-to-pixel level.
- The occluded parts in S cannot be recovered by image-warping; thus, should be inpainted.
- An occlusion map $\hat{\mathcal{O}}_{S\leftarrow D}$ to mask out the feature map regions that should be inpainted.

Method: Occlusion-aware image generation

- In other words, the occlusion mask diminishes the impact of the features corresponding to the occluded parts.

$$\boldsymbol{\xi}' = \hat{\mathcal{O}}_{\mathbf{S}\leftarrow\mathbf{D}} \odot f_w(\boldsymbol{\xi}, \hat{\mathcal{T}}_{\mathbf{S}\leftarrow\mathbf{D}}) \tag{8}$$

where $f_w(\cdot,\cdot)$ denotes the back-warping operation and $\odot$ denotes the Hadamard product. We estimate

Method: Training losses / the reconstruction loss

- Based on the perceptual loss using the pre-trained VGG-19 network.

$$L_{rec}(\hat{\mathbf{D}}, \mathbf{D}) = \sum_{i=1}^{I} \left| N_i(\hat{\mathbf{D}}) - N_i(\mathbf{D}) \right|, \tag{9}$$

where $N_i(\cdot)$ is the i^{th} channel feature extracted from a specific VGG-19 layer and I is the number of feature channels in this layer. Additionally we propose to use this loss on a number of resolutions,

Method: Training losses / the equivariance constraint.

- Training the keypoint detector without any annotations may lead to unstable performance.
- Equivariance constraint is one of the most important factors driving the discovery of unsupervised keypoints.
- It forces the model to learn consistent keypoints w.r.t known geometric transformation, such as TPS.

We assume that an image $\mathbf{X}$ undergoes a known spatial deformation $\mathcal{T}_{\mathbf{X}\leftarrow\mathbf{Y}}$. In this case $\mathcal{T}_{\mathbf{X}\leftarrow\mathbf{Y}}$ can be an affine transformation or a thin plane spline deformation. After this deformation we obtain a new image $\mathbf{Y}$. Now by applying our extended motion estimator to both images, we obtain a set of local approximations for $\mathcal{T}_{\mathbf{X}\leftarrow\mathbf{R}}$ and $\mathcal{T}_{\mathbf{Y}\leftarrow\mathbf{R}}$. The standard equivariance constraint writes as:

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

Method: Training losses / the extended equivariance constraint.

- Since our motion estimator does not only predict the keypoints, but also the Jacobians, we need to extend the equivariance loss including constraints on the Jacobians.

After computing the first order Taylor expansions of both sides, we obtain the following constraints (see derivation details in *Sup. Mat.*):

$$\mathcal{T}_{\mathbf{X}\leftarrow\mathbf{R}}(p_k) \equiv \mathcal{T}_{\mathbf{X}\leftarrow\mathbf{Y}} \circ \mathcal{T}_{\mathbf{Y}\leftarrow\mathbf{R}}(p_k), \tag{11}$$

$$\left(\frac{d}{dp}\mathcal{T}_{\mathbf{X}\leftarrow\mathbf{R}}(p)\Big|_{p=p_k}\right) \equiv \left(\frac{d}{dp}\mathcal{T}_{\mathbf{X}\leftarrow\mathbf{Y}}(p)\Big|_{p=\mathcal{T}_{\mathbf{Y}\leftarrow\mathbf{R}}(p_k)}\right)\left(\frac{d}{dp}\mathcal{T}_{\mathbf{Y}\leftarrow\mathbf{R}}(p)\Big|_{p=p_k}\right), \tag{12}$$

Method: Training losses / the extended equivariance constraint.

- However, implementing Eq. (12) with L1 would force the magnitude of the Jacobians to zero, leading to numerical problems.
- To this end, we reformulate this constraint:

$$\mathbb{1} \equiv \left(\frac{d}{dp} \mathcal{T}_{\mathbf{X} \leftarrow \mathbf{R}}(p) \Big|_{p=p_k} \right)^{-1} \left(\frac{d}{dp} \mathcal{T}_{\mathbf{X} \leftarrow \mathbf{Y}}(p) \Big|_{p=\mathcal{T}_{\mathbf{Y} \leftarrow \mathbf{R}}(p_k)} \right) \left(\frac{d}{dp} \mathcal{T}_{\mathbf{Y} \leftarrow \mathbf{R}}(p) \Big|_{p=p_k} \right), \tag{13}$$

where $\mathbb{1}$ is 2×2 identity matrix. Then, L_1 loss is employed similarly to the keypoint location

Results: Ablation study

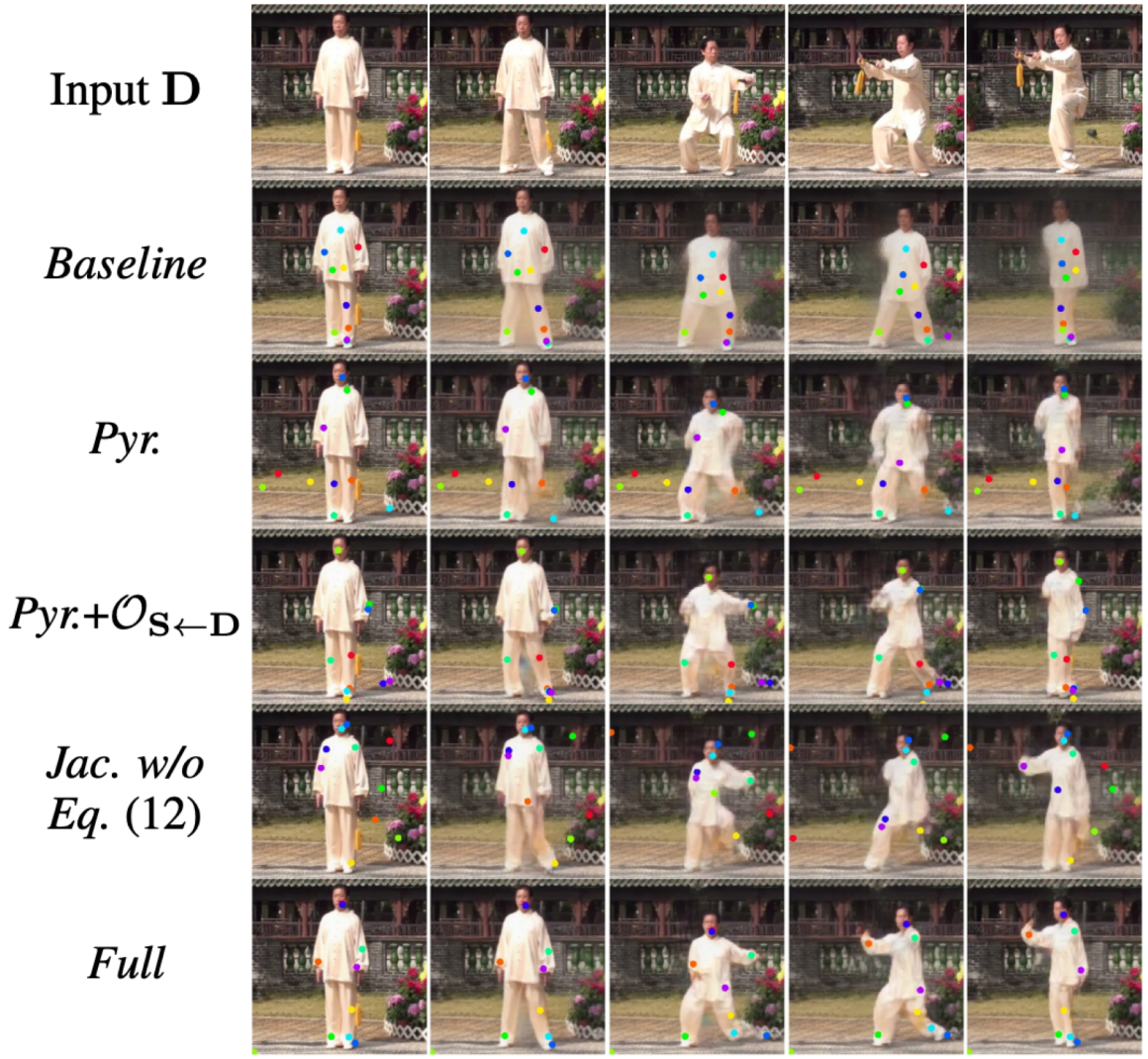


Figure 3: Qualitative ablation on *Tai-Chi-HD*.

Table 1: Quantitative ablation study for video reconstruction on *Tai-Chi-HD*.

	Tai-Chi-HD		
	$\mathcal{L}_1$	(AKD, MKR)	AED
Baseline	0.073	(8.945, 0.099)	0.235
Pyr.	0.069	(9.407, 0.065)	0.213
Pyr.+$\mathcal{O}_{\mathbf{S}\leftarrow\mathbf{D}}$	0.069	(8.773, 0.050)	0.205
Jac. w/o Eq. (12)	0.073	(9.887, 0.052)	0.220
Full	**0.063**	**(6.862, 0.036)**	**0.179**

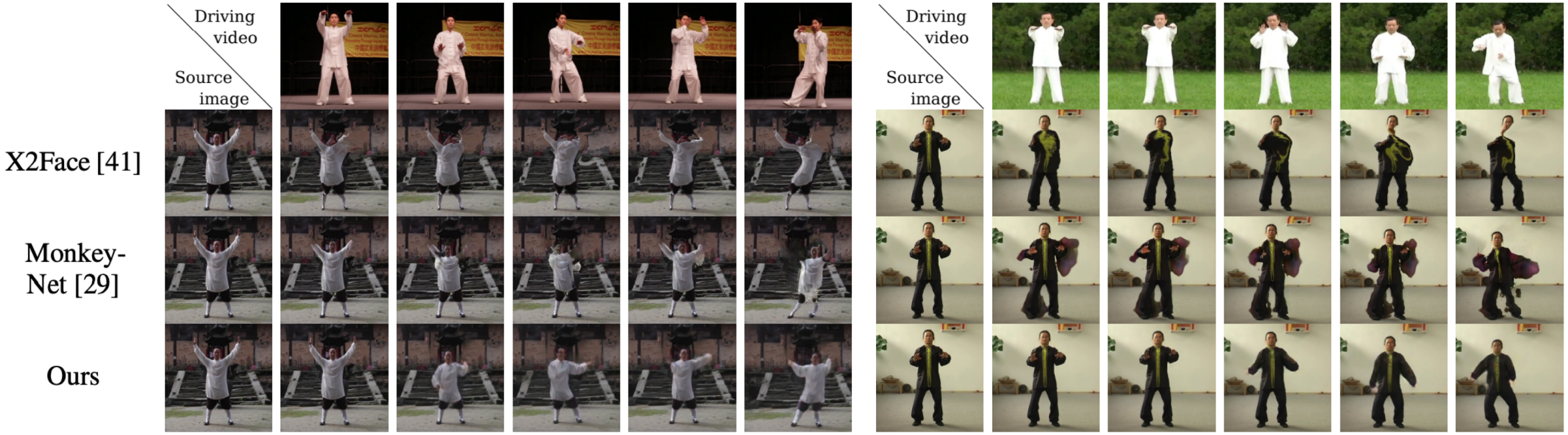
Results: Evaluation metrics

- L1
- Average Keypoint Distance (AKD):
  - For the Tai-Chi-HD, VoxCeleb and Nemo datasets, we use 3rd-party pre-trained keypoint detectors for evaluation.
- Missing Keypoint Rate (MKR):
  - In the case of Tai-Chi-HD, the human-pose estimator returns an additional binary label for each keypoint whether or not the keypoints were successfully detected.
- Average Euclidean Distance (AED):
  - Report the AED between the groudtruth and generated frame representation.

Results: Comparisons.

Table 3: Video reconstruction: comparison with the state of the art on four different datasets.

	Tai-Chi-HD			*VoxCeleb*			*Nemo*			*Bair*
	$\mathcal{L}_1$	(AKD, MKR)	AED	$\mathcal{L}_1$	AKD	AED	$\mathcal{L}_1$	AKD	AED	$\mathcal{L}_1$
X2Face [41]	0.080	(17.654, 0.109)	0.272	0.078	7.687	0.405	0.031	3.539	0.221	0.065
Monkey-Net [29]	0.077	(10.798, 0.059)	0.228	0.049	1.878	0.199	0.018	1.285	0.077	0.034
Ours	**0.063**	**(6.862, 0.036)**	**0.179**	**0.043**	**1.294**	**0.140**	**0.016**	**1.119**	**0.048**	**0.027**



Results: keypoint detector

Thank you !

Deep Video Generation vs Object (Image) Animation

- Deep Video Generation:
  - This task usually takes conditional information as input that comprises **categoriacal labels** or **static images** and produces video frames of desired actions.
- Object (Image) Animation / Video Re-targeting:
  - Image Animation from a driving video can be interpreted as the problem of *transferring motion information* from one domain to another.

Deep Video Generation vs Object (Image) Animation

- Deep Video Generation:
  - This task usually takes conditional information as input that comprises **categoriacal labels** or **static images** and produces video frames of desired actions.

It is a more challenging task since image animation requires **decoupling motion and content information**, as well as a recombining them.

- Object (Image) Animation / Video Re-targeting :
  - Image Animation from a driving video can be interpreted as the problem of *transferring motion information* from one domain to another.

Previous work: Image Animation

- Displaced dynamic expression regression for real-time facial tracking and animation, TOG’14
- Face2face: Real-time face capture and reenactment of rgb videos, CVPR’16
- Recycle-GAN: Unsupervised video retargeting, ECCV’18
- Everybody dance now, ECCV’18
- X2face: A network for controlling face generation using image, audio, and pose codes, ECCV’18
- Animating Arbitary Object via Deep Motion Transfer, CVPR’19
- First Order Motion Model for Image Animation, Neurips’19

Previous work: Image Animation

- Displaced dynamic expression regression for real-time facial tracking and animation, TOG'14
- Face2face: Real-time face capture and reenactment of rgb videos, CVPR'16

- Recycle-GAN: Unsupervised video retargeting, ECCV'18
- Everybody dance now, ECCV'18

- X2face: A network for controlling face generation using image, audio, and pose codes, ECCV'18
- Animating Arbitary Object via Deep Motion Transfer, CVPR'19
- First Order Motion Model for Image Animation, Neurips'19

Animating Arbitary Object via Deep Motion Transfer, CVPR'19

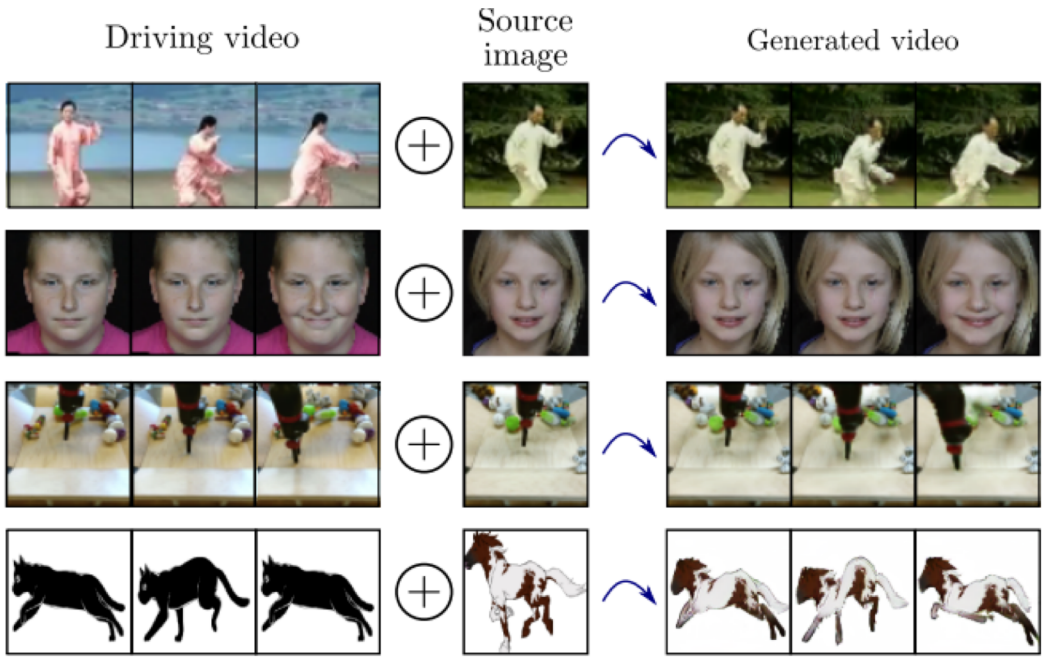


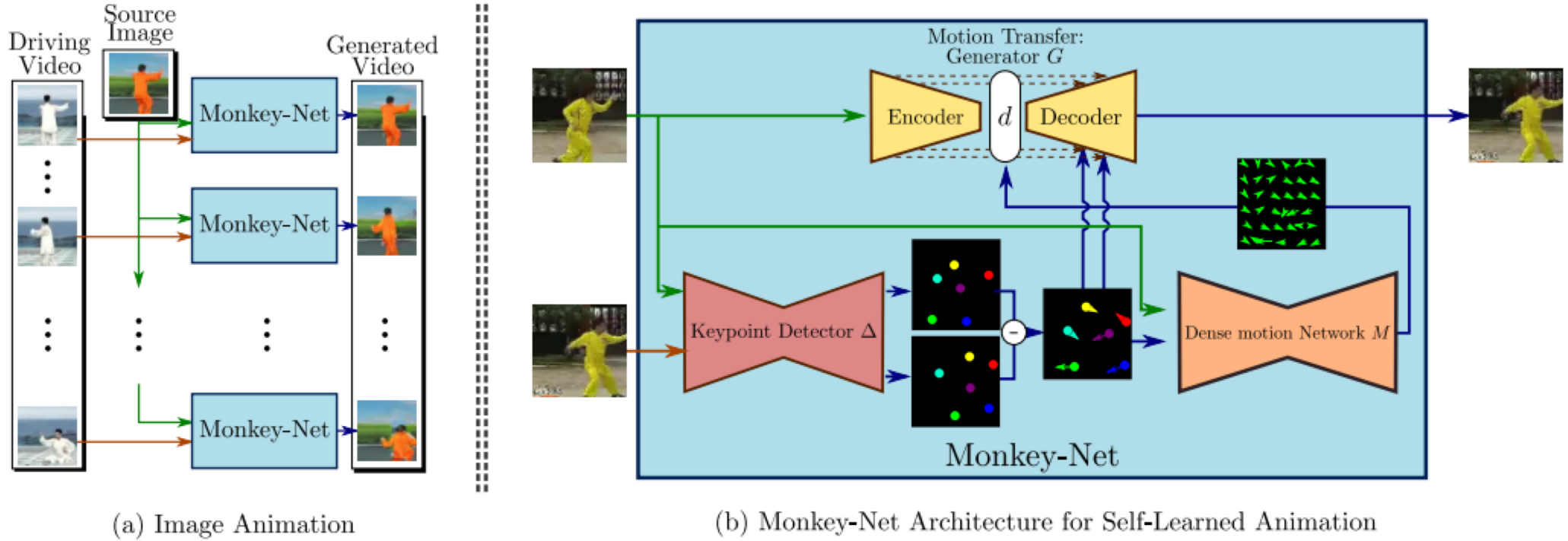
Figure 1: Our deep motion transfer approach can animate arbitrary objects following the motion of the driving video.

- The objective of this work is to animate an object based on the motion of a similar object in a driving video.

Animating Arbitary Object via Deep Motion Transfer, CVPR'19

- Authors propose learning a latent representation of an object category in a self-supervised way.
- This approach is not designed for specific object category, but rather is effective in animating arbitrary objects.

Animating Arbitary Object via Deep Motion Transfer, CVPR'19



(a) Image Animation

(b) Monkey-Net Architecture for Self-Learned Animation

Figure 2: A schematic representation of the proposed motion transfer framework for image animation. At testing time (Fig. (a)), the model generates a video with the object appearance of the source image but with the motion from the driving video. Monkey-Net (Fig. (b)) is composed of three networks: a motion-specific keypoint detector Δ, a motion prediction network M and an image generator G. G reconstructs the image $\boldsymbol{x}'$ from the keypoint positions $\Delta(\boldsymbol{x})$ and $\Delta(\boldsymbol{x}')$. The optical flow computed by M is used by G to handle misalignments between $\boldsymbol{x}$ and $\boldsymbol{x}'$. The model is learned with a self-supervised learning scheme.

Animating Arbitary Object via Deep Motion Transfer, CVPR'19

- 1) Keypoint Detector Δ:
- 2) Dense Motion prediction network M:
- 3) Motion Transfer network G:

Animating Arbitary Object via Deep Motion Transfer, CVPR'19

- 1) Keypoint Detector Δ:

$$\boldsymbol{h}_k = \sum_{p \in \mathcal{U}} H_k[p]p;\ \Sigma_k = \sum_{p \in \mathcal{U}} H_k[p](p - \boldsymbol{h}_k)(p - \boldsymbol{h}_k)^\top \quad (1)$$

  - Unsupervised keypoint detection inspired by
  - Estimating K heatmaps $H_k \in [0,1]^{H \times W}$, one for each keypoint
  - To model the keypoint location confidence, authors fit a Gaussian on each detection confidence map.

$$\forall p \in \mathcal{U}, H_k(\mathbf{p}) = \frac{1}{\alpha} exp\left(-(\mathbf{p} - \boldsymbol{h}_k)\Sigma_k^{-1}(\mathbf{p} - \boldsymbol{h}_k)\right) \quad (2)$$

  - The keypoint detector Δ is trained with a generator G together according to the following objective: G should be able to reconstruct x' from the keypoint location $\Delta(x)$, $\Delta(x')$, and x

Animating Arbitary Object via Deep Motion Transfer, CVPR'19

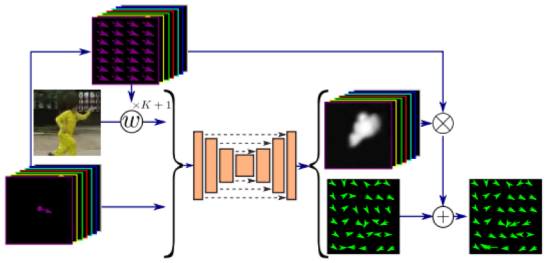


Figure 3: A schematic representation of the adopted part-based model for optical flow estimation from sparse representation. From the appearance of the first frame and the keypoints motion, the network M predicts a mask for each keypoint and the residual motion (see text for details).

$$\mathcal{F}_{\text{coarse}} = \sum_{k=1}^{K+1} M_k \otimes \rho(h_k)$$

$$\mathcal{F} = \mathcal{F}_{\text{coarse}} + \mathcal{F}_{\text{residual}}$$

- 2) Dense Motion prediction network M (from sparse keypoints to desne optical flow):
  - The task of predicting a **dense optical flow** only from the displacement of a few keypoints and the appearance of the first frame is challenging.
  - First, estimating masks $M_k \in \mathbb{R}^{H \times W}$ that segment the object in rigid parts corresponding to each keypoints
  - $\rho(\cdot) \in \mathbb{R}^{H \times W \times 2}$ is the operator that returns a tensor by repeating the input vector H x W times.
  - Since a standard U-net cannot handle large pixe2pixel misalignment between the input and the output images, a deformation module (warping function $f_w(\cdot,\cdot)$) is proposed according to $\mathcal{F}$.

Animating Arbitary Object via Deep Motion Transfer, CVPR'19

- Network Training:
  - Adversarial loss
    - $$\mathcal{L}^{D}_{\text{gan}}(D) = \mathbb{E}_{\boldsymbol{x}' \in \mathcal{X}}[(D(\boldsymbol{x}' \oplus H') - 1)^2] + \mathbb{E}_{(\boldsymbol{x}, \boldsymbol{x}') \in \mathcal{X}^2}[D(\hat{\boldsymbol{x}}' \oplus H'))^2]$$
    - $$\mathcal{L}^{G}_{\text{gan}}(G) = \mathbb{E}_{(\boldsymbol{x}, \boldsymbol{x}') \in \mathcal{X}^2}[(D(\hat{\boldsymbol{x}}' \oplus H') - 1)^2]$$
  - Feature matching loss
    - $$\mathcal{L}_{\text{rec}} = \mathbb{E}_{(\boldsymbol{x}, \boldsymbol{x}')}\left[\|D_i(\hat{\boldsymbol{x}}' \oplus H') - D_i(\boldsymbol{x}' \oplus H'))\|_1\right]$$
  - Total loss
    - $$\mathcal{L}_{\text{tot}} = \lambda_{\text{rec}}\mathcal{L}_{\text{rec}} + \mathcal{L}^{G}_{\text{gan}}$$

Animating Arbitary Object via Deep Motion Transfer, CVPR'19

- Generation Procedure:
  - At test time, the network receives a driving video and a source image.
  - In order to generate t^{th} frame, Δ estimates the keypoint locations h_k^s in the source image.
  - Similarly, it estimates the keypoint locations h_k^1 and h_k^t from first and the t^{th} frames of the driving video.
  - The keypoints in the generated frame are given by:

$$h_k^{s\prime} = h_k^s + (h_k^t - h_k^1)$$

  - And then, they are encoded as heatmaps using the covariance matrices.
  - Finally, the heatmaps are given to the dense motion M and the generator G together with the source image.

Animating Arbitary Object via Deep Motion Transfer, CVPR'19

- Results:

Figure 5: Qualitative results for image animation on the Nemo dataset: X2face (2-nd row) against our method (3-rd row).

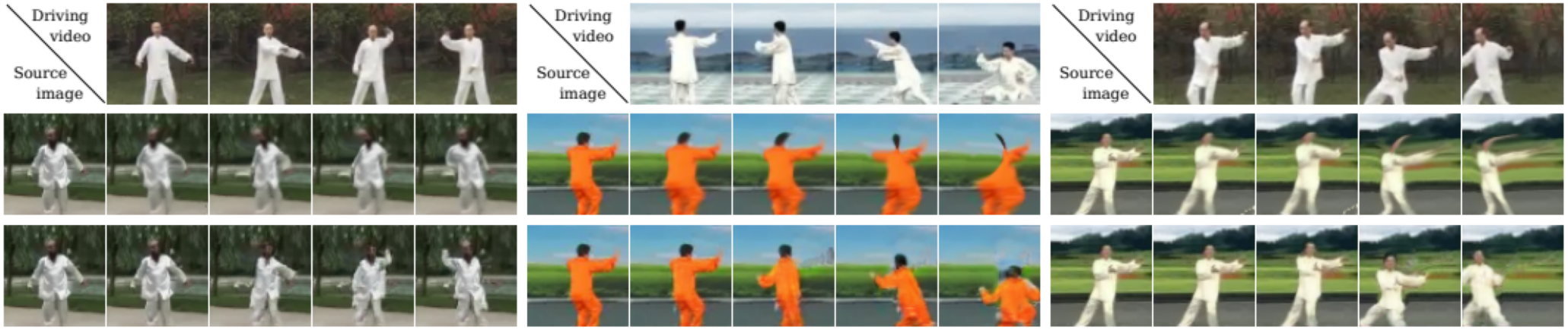


Figure 6: Qualitative results for image animation on the *Tai-Chi* dataset: X2face (2-nd row) against our method (3-rd row)

Animating Arbitary Object via Deep Motion Transfer, CVPR'19

- Metric:
  - **L1** : In the case of the video reconstruction task where the ground truth video is available, we can compare the average L1 distance between frames.
  - **Average Keypoint Distance (AKD)** : In order to evaluate whether the motion of the generated video, authors employ externally pre-trained human-pose estimator for Tai-Chi dataset and the facial landmark detector for Nemo dataset
  - **Missing Keypoint Rate (MKR)** : In the case of the Tai-Chi dataset, the pre-trained human-pose estimator returns also a binary label for each keypoint indicating whether the keypoints were successfully detected. MKR is the percentage of keypoints that are detected from the ground-truth frame but not in the generated one.

Animating Arbitary Object via Deep Motion Transfer, CVPR'19

- Metric:
  - **Average Euclidean Distance (AED)** : AED is the feature-based metric used in [1] that consists in computing the distance between a feature representation of the ground-truth and the generated video frames. Authors employ the person re-id network for Tai-Chi and the facial identification network for Nemo.
  - **Frechet Inception Distance (FID)** : To evaluate the quality of individual frames
  - **User study**

[1] A variational u-net for conditional appearance and shape generation, CVPR'18

Animating Arbitary Object via Deep Motion Transfer, CVPR'19

- Quantitative Results:

	Tai-Chi			Nemo			Bair
	$\mathcal{L}_1$	(AKD, MKR)	AED	$\mathcal{L}_1$	AKD	AED	$\mathcal{L}_1$
X2Face	0.068	(4.50, 35.7%)	0.27	0.022	0.47	0.140	0.069
Ours	**0.050**	**(2.53, 17.4%)**	**0.21**	**0.017**	**0.37**	**0.072**	**0.025**

Table 1: Video reconstruction comparisons

Tai-Chi	*Nemo*	*Bair*
85.0%	79.2%	90.8%

Table 4: User study results on image animation. Proportion of times our approach is preferred over X2face [50].

Tai-Chi			
	FID	AED	MKR
MoCoGAN [43]	54.83	0.27	46.2%
Ours	**19.75**	**0.17**	**30.3%**

Nemo		
	FID	AED
MoCoGAN [43]	51.50	0.33
CMM-Net [49]	27.27	0.13
Ours	**11.97**	**0.12**

Bair	
	FID
MoCoGAN [43]	244.00
SV2P [2]	57.90
Ours	**23.20**

Table 3: Image-to-video translation comparisons.

Animating Arbitary Object via Deep Motion Transfer, CVPR'19

- Ablation Study:

	Tai-Chi		
	$\mathcal{L}_1$	(AKD, MKR)	AED
No $\mathcal{F}$	0.057	(3.11, 23.8%)	0.24
No $\mathcal{F}_{\text{residual}}$	0.051	(2.81, 18.0%)	0.22
No $\mathcal{F}_{coarse}$	0.052	(2.75, 19.7%)	0.22
No Σ_k	0.054	(2.86, 20.6%)	0.23
No $\boldsymbol{x}$	0.051	(2.71, 19.3%)	**0.21**
Full	**0.050**	**(2.53, 17.4%)**	**0.21**

Table 2: Video reconstruction ablation study *TaiChi*.

Figure 4: Qualitative ablation evaluation of video reconstruction on *Tai-Chi*.

Datasets:

- Previously used for video generation:
  - Tai-Chi dataset:
    - Clips downloaded from YouTube composed of 4500 video
    - The video length varies from 32 to 100 frames.
  - BAIR robot pushing:
    - Videos collected by a Sawyer robotic arm pushing a variety of objects over a table
    - It contains 40960 and 256 videos for train and test respectively.
  - UvA-NEMO Smile dataset:
    - a facial dynamics composed of 1240 video
    - Faces are aligned using the OpenFace library and each video has 32 frames.