# Emergent Correspondence from Image Diffusion

Luming Tang*

Menglin Jia*

Qianqian Wang*

Cheng Perng Phoo

Bharath Hariharan

NEURAL INFORMATION PROCESSING SYSTEMS

Cornell University

*Equal Contribution

# TL; DR

Correspondences emerge from image diffusion models *without any explicit supervision*.

# Visual Correspondence

- A fundamental problem in computer vision
- Critical building block for many tasks

3D reconstruction

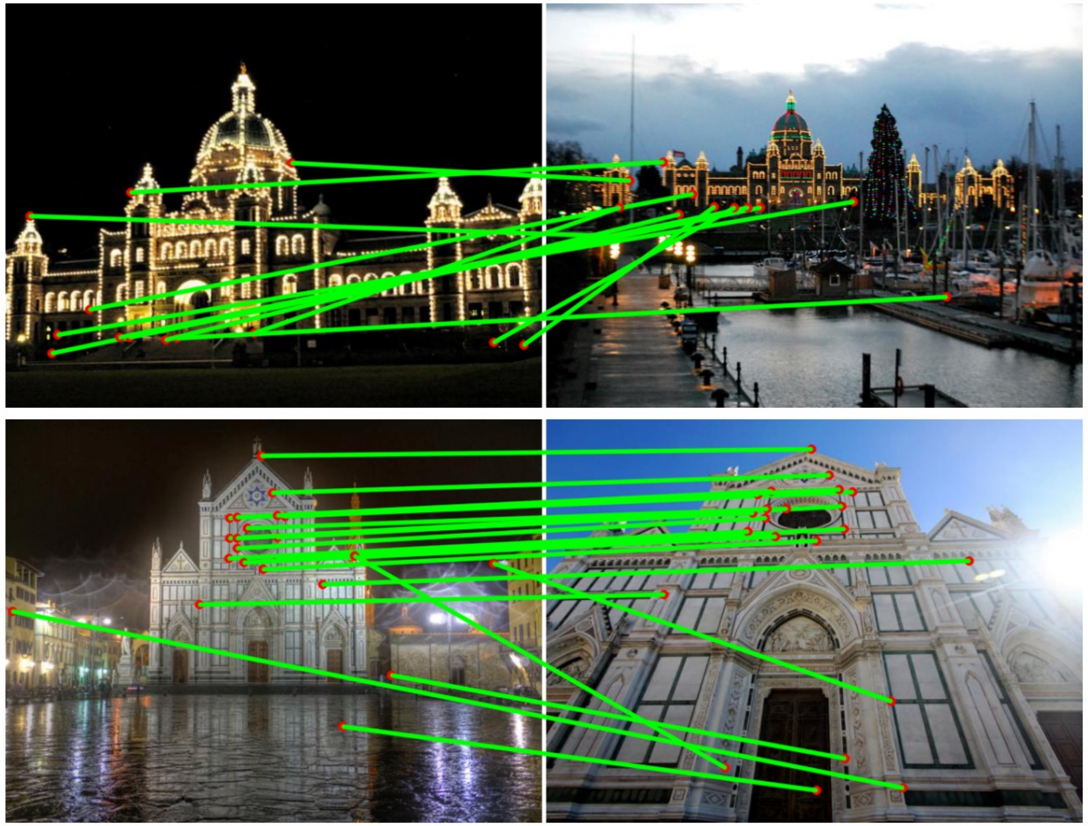
image editing

video tracking

https://www.cs.cornell.edu/~snavely/bundler; https://neural-congealing.github.io; https://co-tracker.github.io

# Visual Correspondence

- Easy for humans: learned without any correspondence labels



match object parts across
viewpoint and lighting changes

https://qianqianwang68.github.io/CAPS

# Visual Correspondence

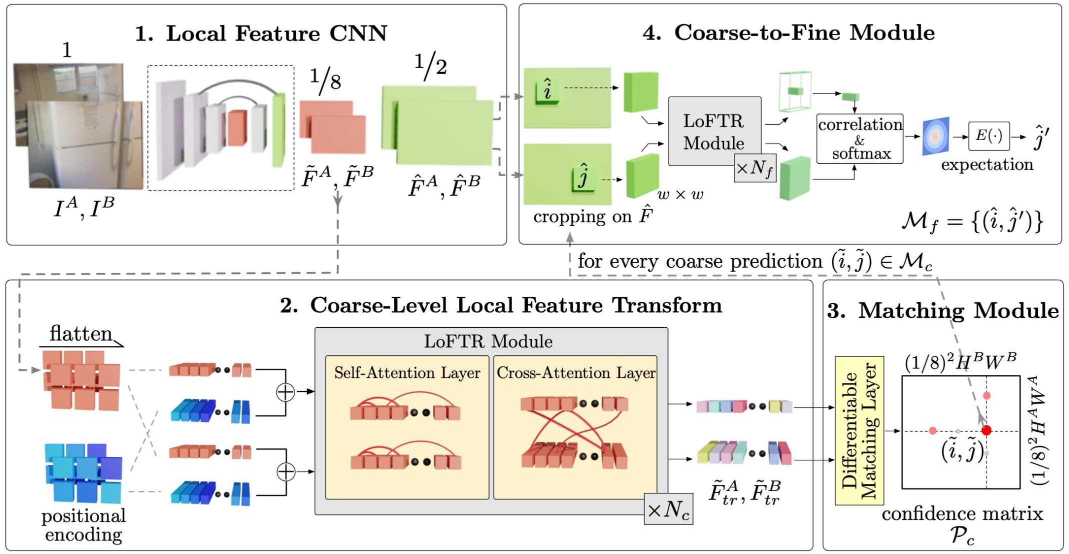
- Easy for humans: learned without any correspondence labels

match object parts across
instances, categories,
and even modalities

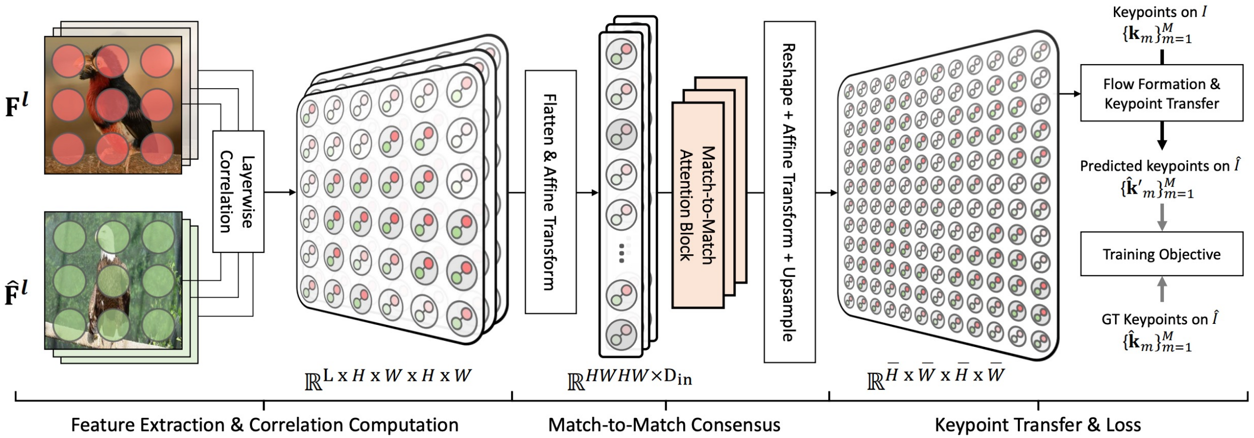
https://unsplash.com

# Visual Correspondence

- Easy for humans: learned without any correspondence labels
- Hard for computers: need to train with large-scale labeled data




LoFTR: Detector-Free Local Feature Matching with Transformers
[Sun et al. 2021]




TransforMatcher: Match-to-Match Attention for Semantic Correspondence
[Kim et al. 2022]

# Visual Correspondence

- Easy for humans: learned without any correspondence labels
- Hard for computers: need to train with large-scale labeled data

Can computer vision system similarly learn
accurate correspondences **without** labeled data?

# Diffusion Models

- A family of generative models with superior image generation capabilities.

images generated by
Stable Diffusion

https://github.com/Stability-AI/stablediffusion

# Diffusion Models

- Superior image generation capacities.
- Also enable image-to-image translation.

Implicit Correspondence?

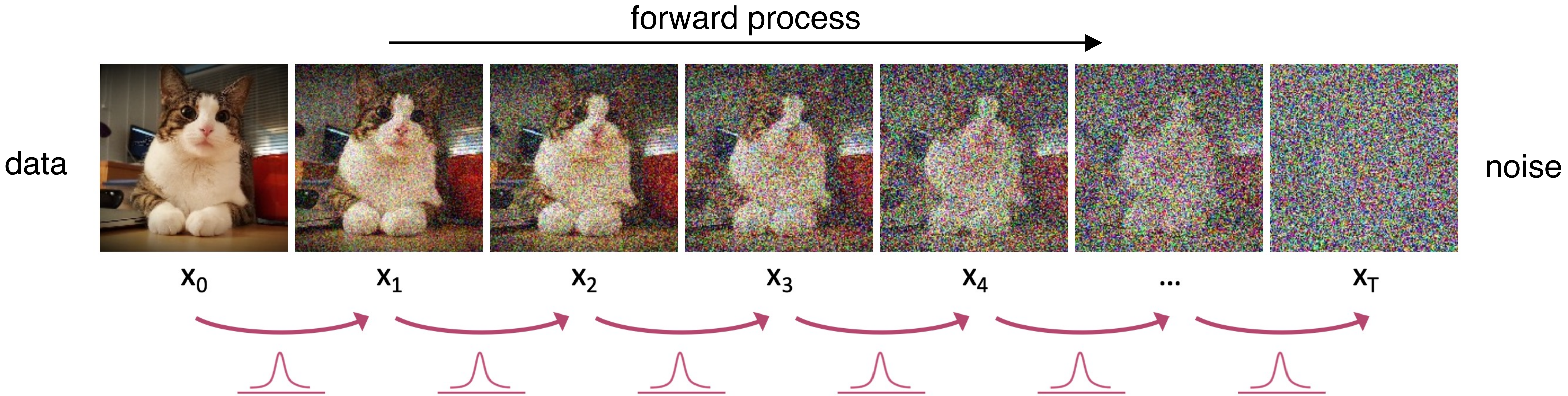
Cat → Dog

https://pix2pixzero.github.io

# Diffusion Models

- Learning to **generate** by **denoising**
  - Forward process gradually add Gaussian noise to input data

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\,\epsilon \qquad \text{where} \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$




Figures modified from https://cvpr2023-tutorial-diffusion-models.github.io

# Diffusion Models

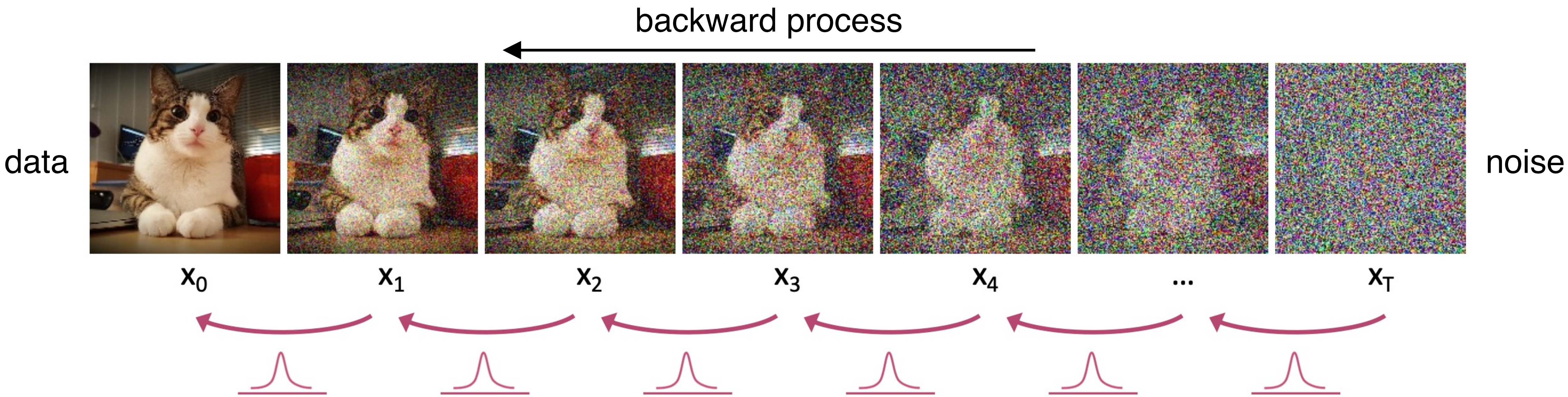
- Learning to **generate** by **denoising**
    - Forward process gradually add Gaussian noise to input data
    - Backward process **learn** to **predict and remove noise**

$$\epsilon_\theta(x_t, t) \rightarrow \epsilon$$

# Diffusion Models

- Learning to **generate** by **denoising**
  - Forward process gradually add Gaussian noise to input data
  - Backward process **learn** to **predict and remove noise**

$$\epsilon_\theta(x_t, t) \rightarrow \epsilon$$



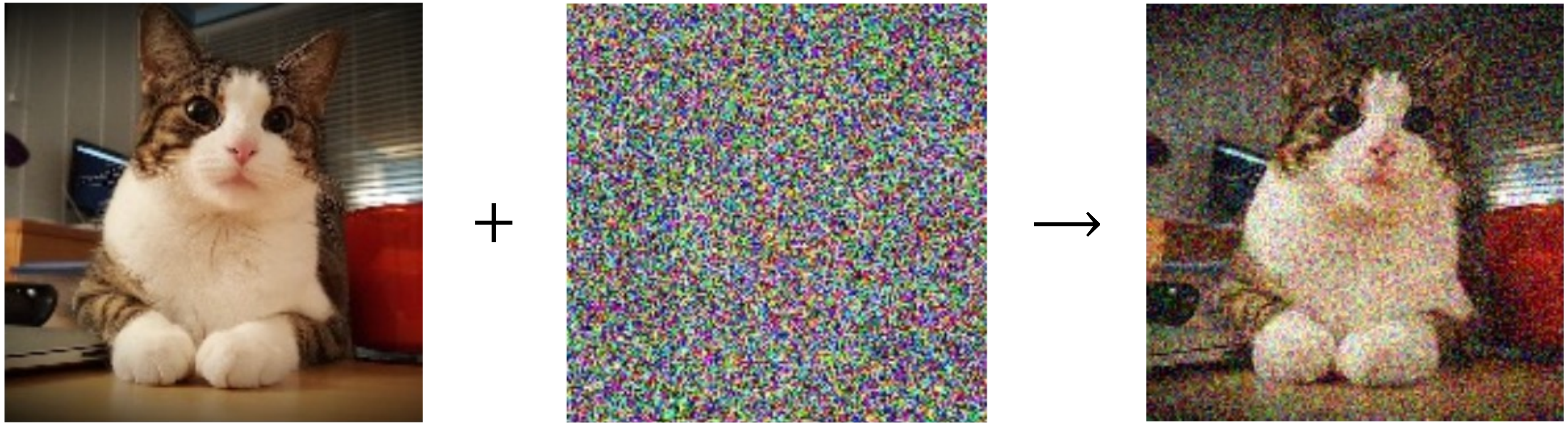

Figures modified from https://cvpr2023-tutorial-diffusion-models.github.io

# Diffusion Features

Given an input image, we extract image features using a pre-trained diffusion model, then use them to do correspondence tasks.

# Diffusion Features

1. Add noise to input image $x_0$ to get $x_t$

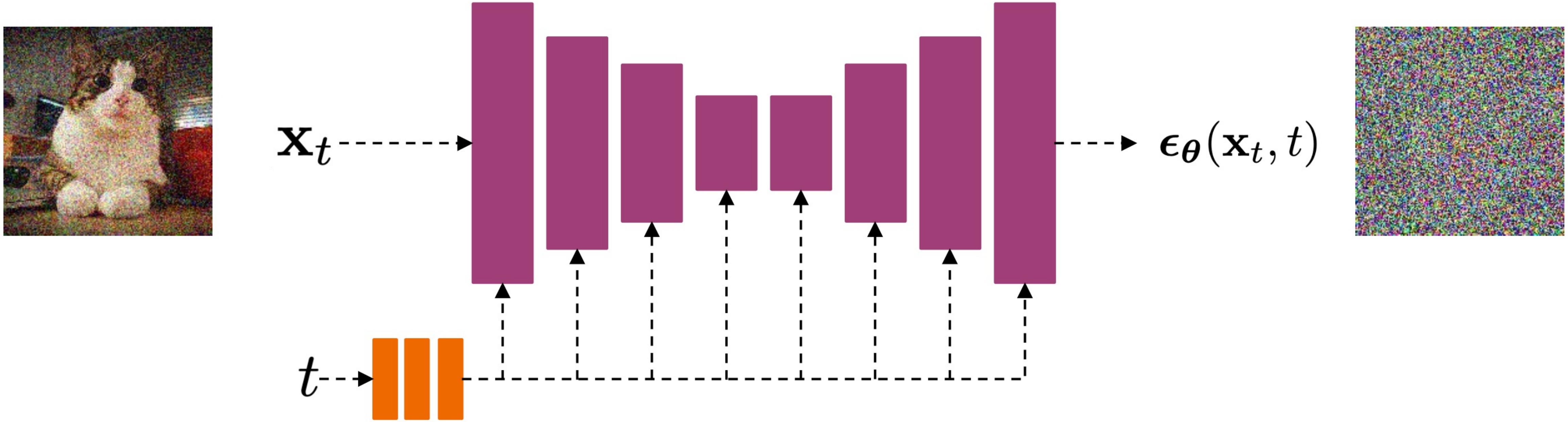


$x_0$ $\epsilon$ $x_t$

Figures modified from https://cvpr2023-tutorial-diffusion-models.github.io

# Diffusion Features

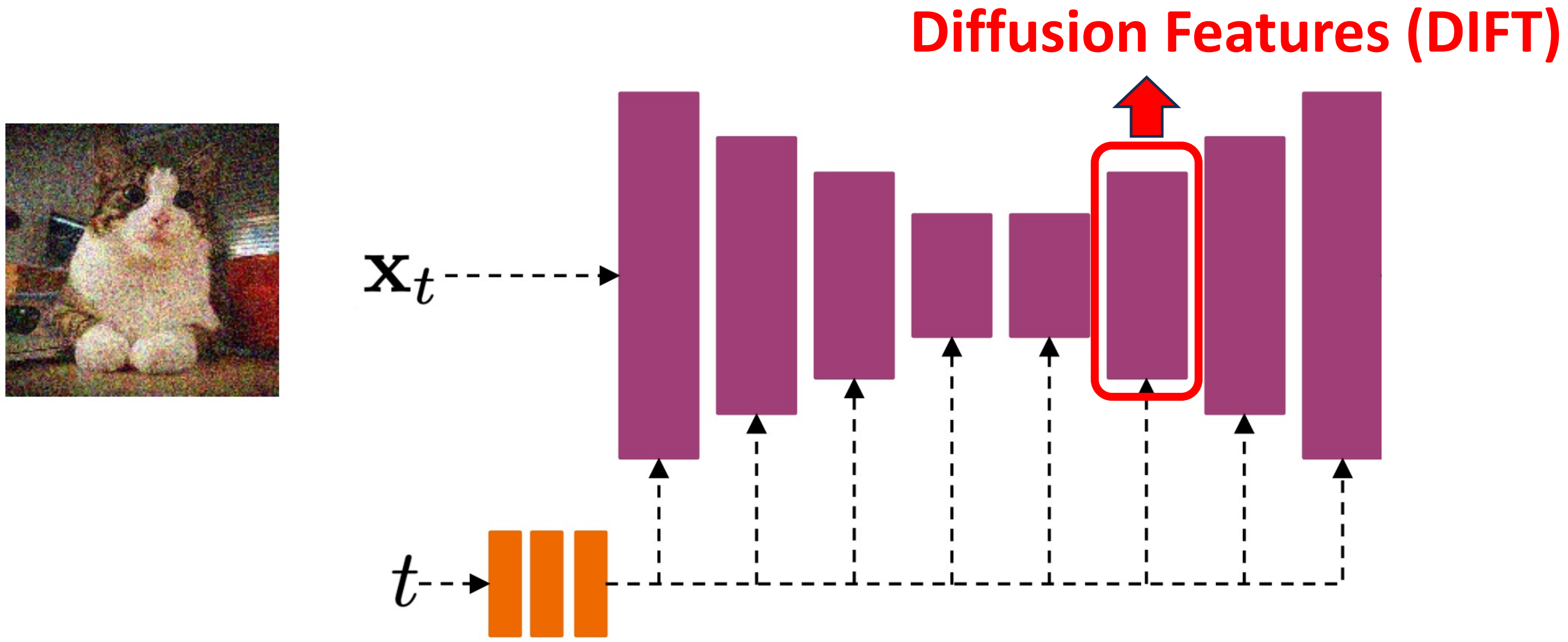
1. Add noise to input image $x_0$ to get $x_t$
2. Feed $x_t$ and ***t*** into the denoising network $\epsilon_\theta(x_t, t)$




Figures modified from https://cvpr2023-tutorial-diffusion-models.github.io

# Diffusion Features (DIFT)

1. Add noise to input image $x_0$ to get $x_t$
2. Feed $x_t$ and ***t*** into the denoising network $\epsilon_\theta(x_t, t)$
3. Get intermediate feature maps at certain block ***i*** as DIFT




Figures modified from https://cvpr2023-tutorial-diffusion-models.github.io

# Diffusion Features (DIFT)

1. Add noise to input image $x_0$ to get $x_t$
2. Feed $x_t$ and ***t*** into the denoising network $\epsilon_\theta(x_t, t)$
3. Get intermediate feature maps at certain block ***i*** as DIFT
4. Interpolate the feature map to get each point's feature vector

# Diffusion Features (DIFT)

1. Add noise to input image $x_0$ to get $x_t$
2. Feed $x_t$ and ***t*** into the denoising network $\epsilon_\theta(x_t, t)$
3. Get intermediate feature maps at certain block ***i*** as DIFT
4. Interpolate the feature map to get each point's feature vector
5. Feature matching using cosine similarity to get correspondences

No Extra Training Needed

# DIFT can do semantic correspondence

- Without any explicit supervision, DIFT can find correspondences on real images across instances, categories, and even domains.

# DIFT can do semantic correspondence

- Without any explicit supervision, DIFT can find correspondences on real images across instances, categories, and even domains.

# DIFT can do semantic correspondence

- Eval Dataset: SPair, 12k image pairs on 18 categories

- Baselines:

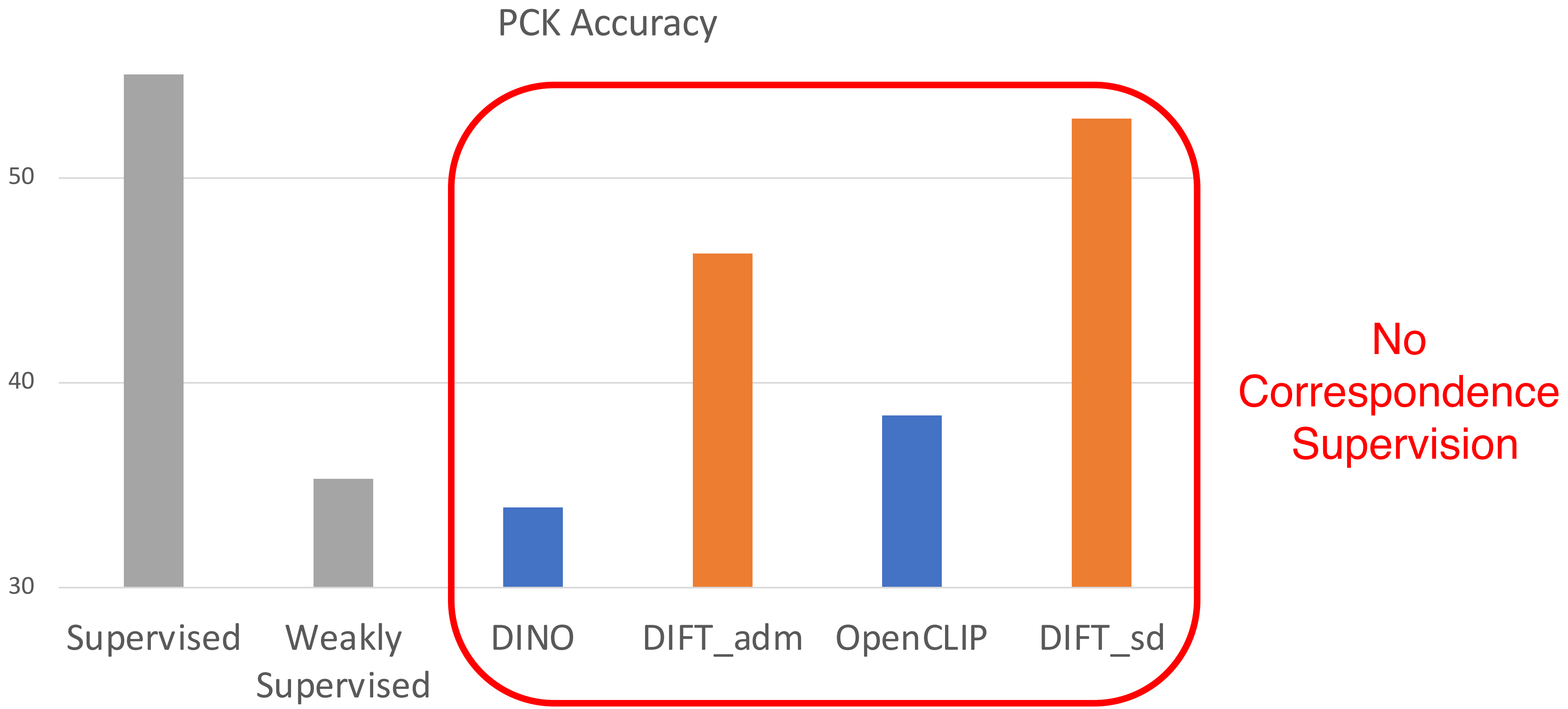
  a) Supervised Methods: trained with correspondence labels

  b) Weakly-Supervised Methods: trained with image pairs

  c) Off-the-shelf self-supervised features: DINO, OpenCLIP

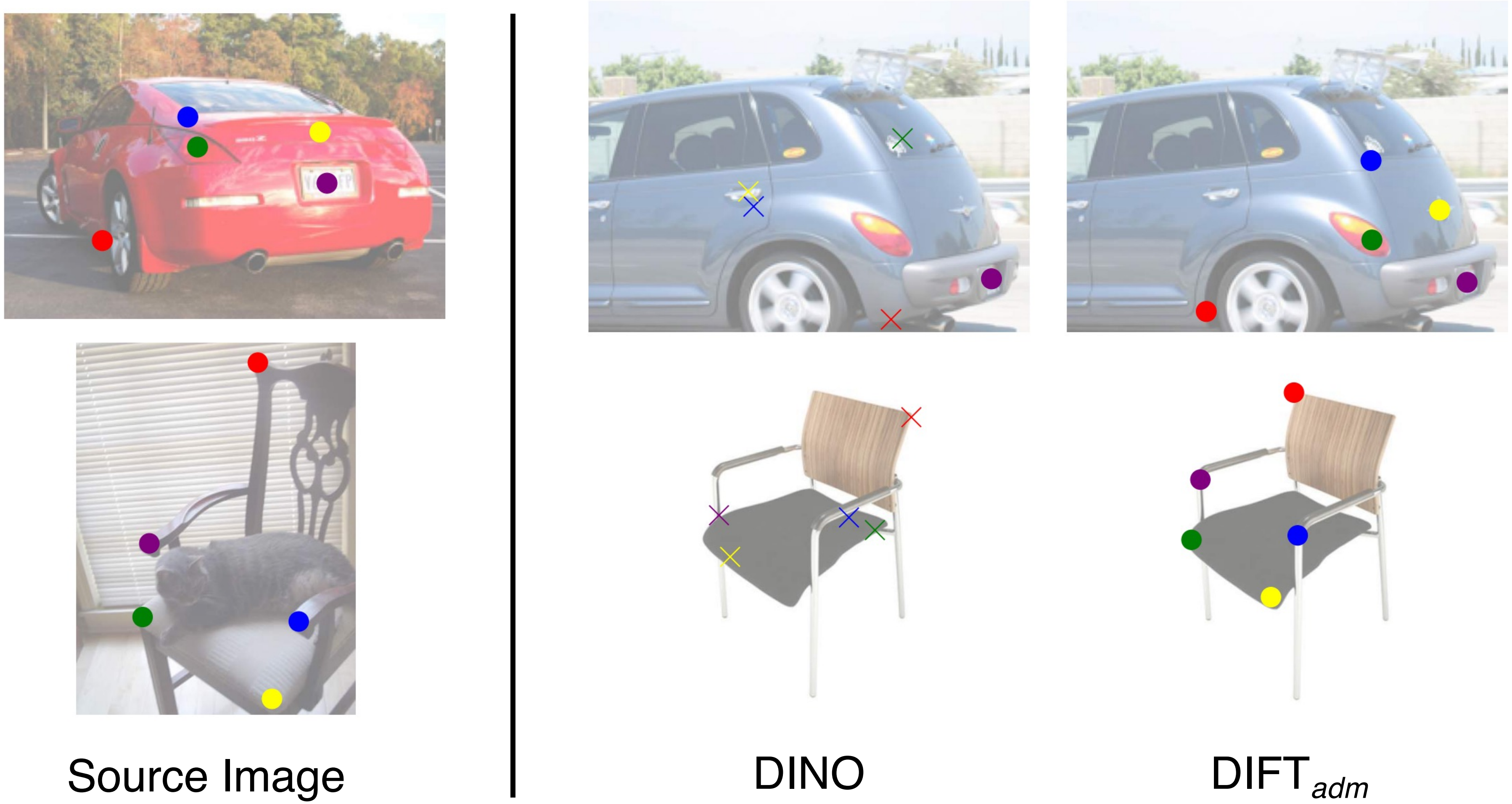
# DIFT can do semantic correspondence

- DIFT outperforms self-supervised features and weakly-supervised methods with large margin, even on par with SOTA supervised methods.



# DIFT can do semantic correspondence

- DINO vs. $\text{DIFT}_{adm}$ : both trained on ImageNet without class labels



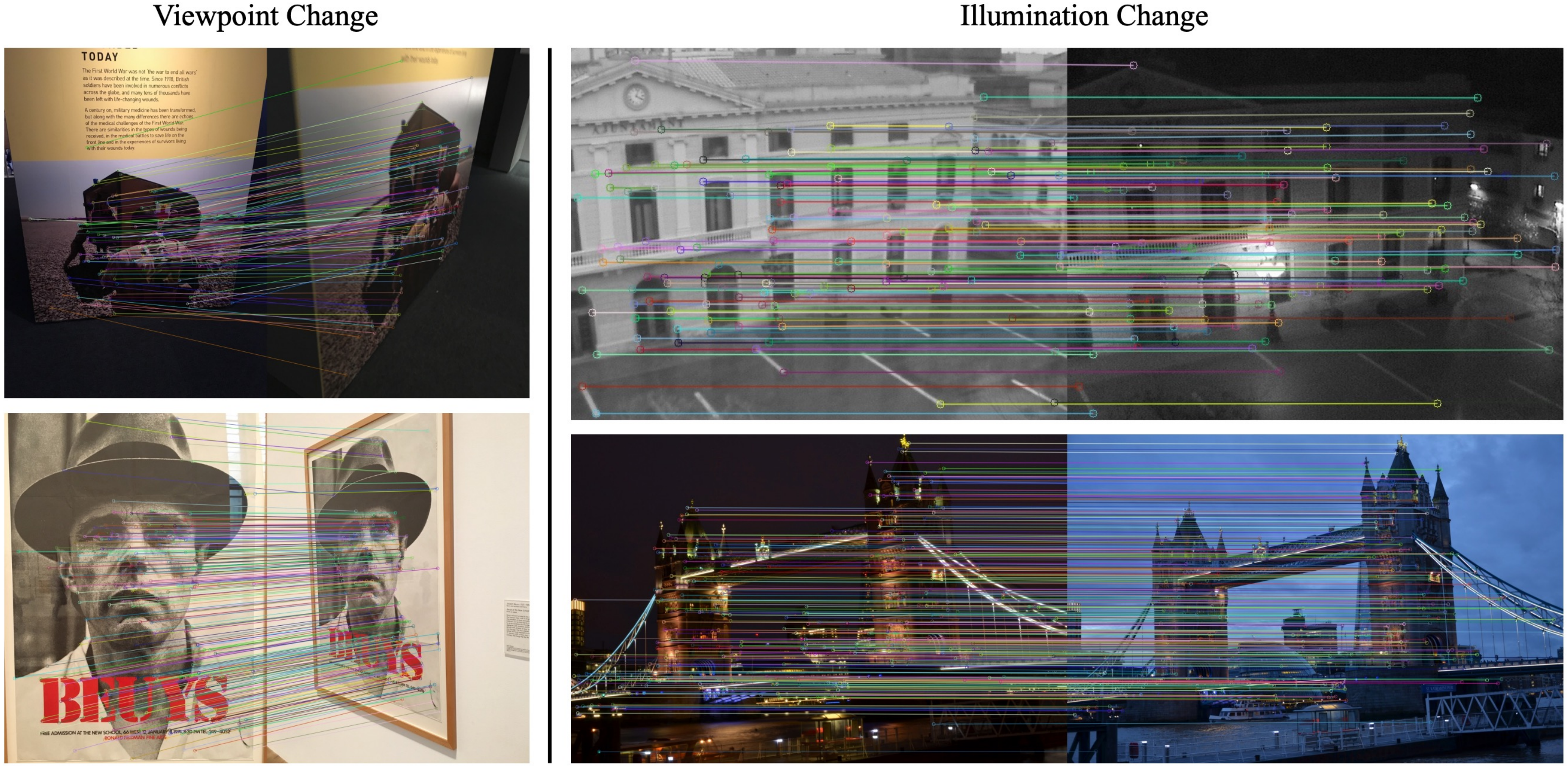
Source Image | DINO | $\text{DIFT}_{adm}$

# DIFT can do semantic correspondence

- DIFT can propagate edits from one image to others across different instances, categories, and domains.

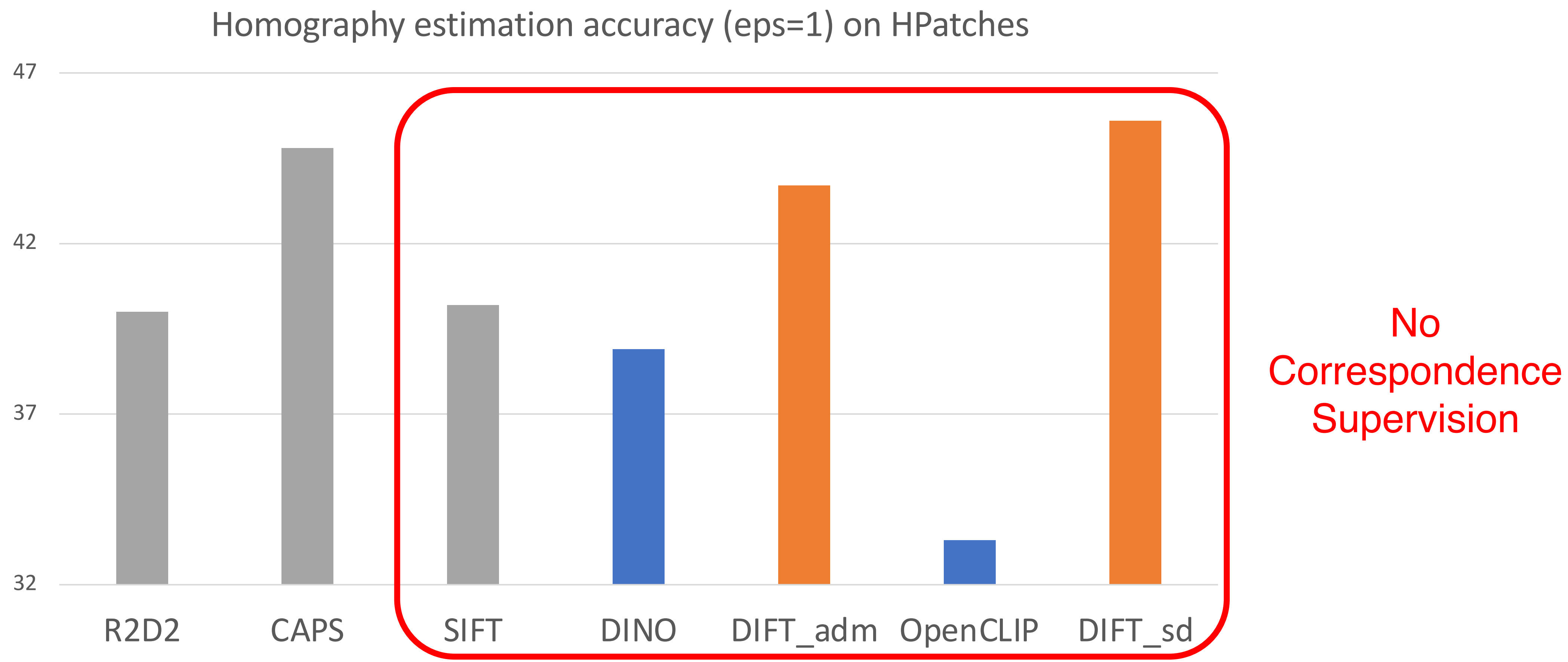
# DIFT can do geometric correspondence

- DIFT shows competitive performance on finding geometric correspondences, i.e., image matching and homography estimation.



# DIFT can do geometric correspondence

- DIFT shows competitive performance on finding geometric correspondences, i.e., image matching and homography estimation.



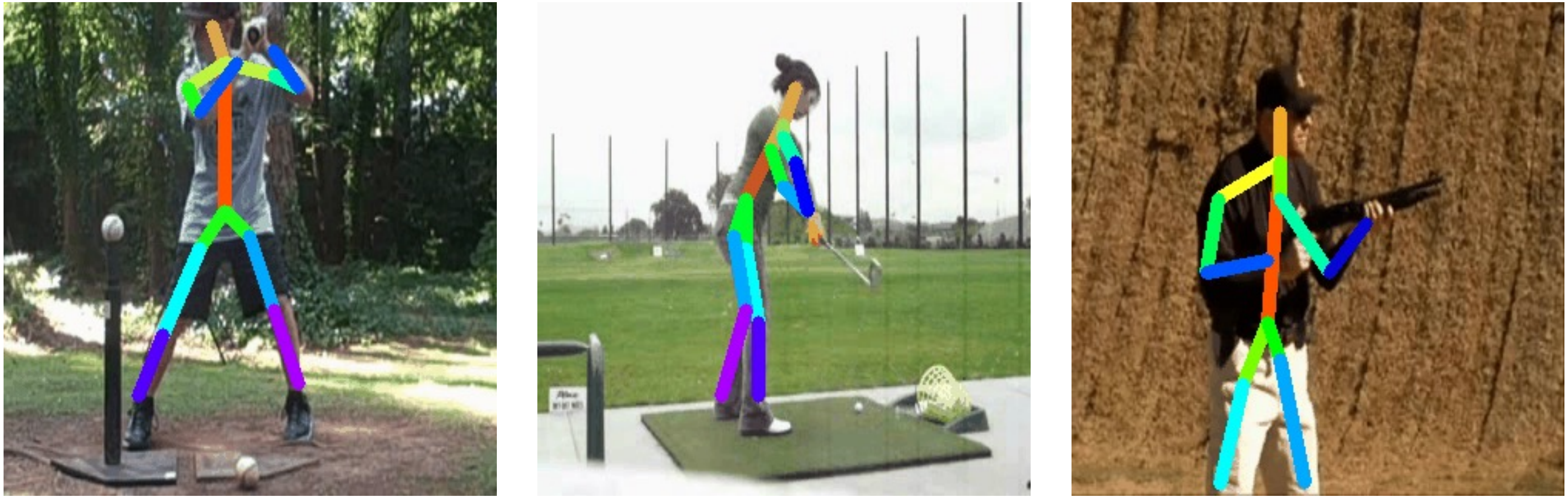
# DIFT can do temporal correspondence

- DIFT demonstrates strong performance on temporal correspondence tasks, although never trained or fine-tuned on video data.

object segmentation using DIFT

# DIFT can do temporal correspondence

- DIFT demonstrates strong performance on temporal correspondence tasks, although never trained or fine-tuned on video data.



pose tracking using DIFT

# Thank you for watching!

More visualizations and the interactive demo at our project page:

https://diffusionfeatures.github.io

source image

target image

source image

target image