DragDiffusion: Harnessing Diffusion Models for Interactive Point-based Image Editing

Yujun Shi[1] Chuhui Xue[2] Jun Hao Liew[2] Jiachun Pan[1]
Hanshu Yan[2] Wenqing Zhang[2] Vincent Y. F. Tan[1] Song Bai[2]
[1]National University of Singapore [2] ByteDance Inc.

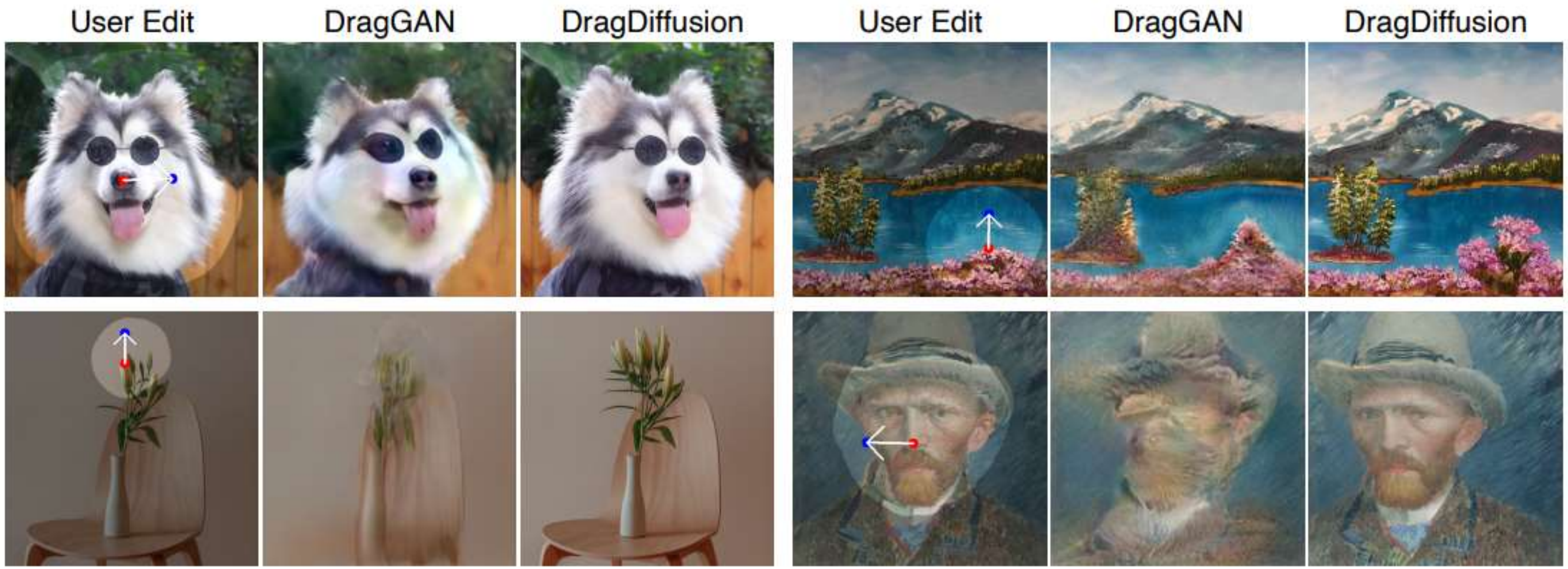
arXiv(CVPR), 2023(2024), 60 citation

Interactive point-based image editing

- Move several handle points to target points
- Requirements of task
  - Flexibility: adjust spatial attributes (e.g., pose, position, expression, shape, etc)
  - Precision: control spatial attributes with high precision
  - Generality: apply various categories

Handle point

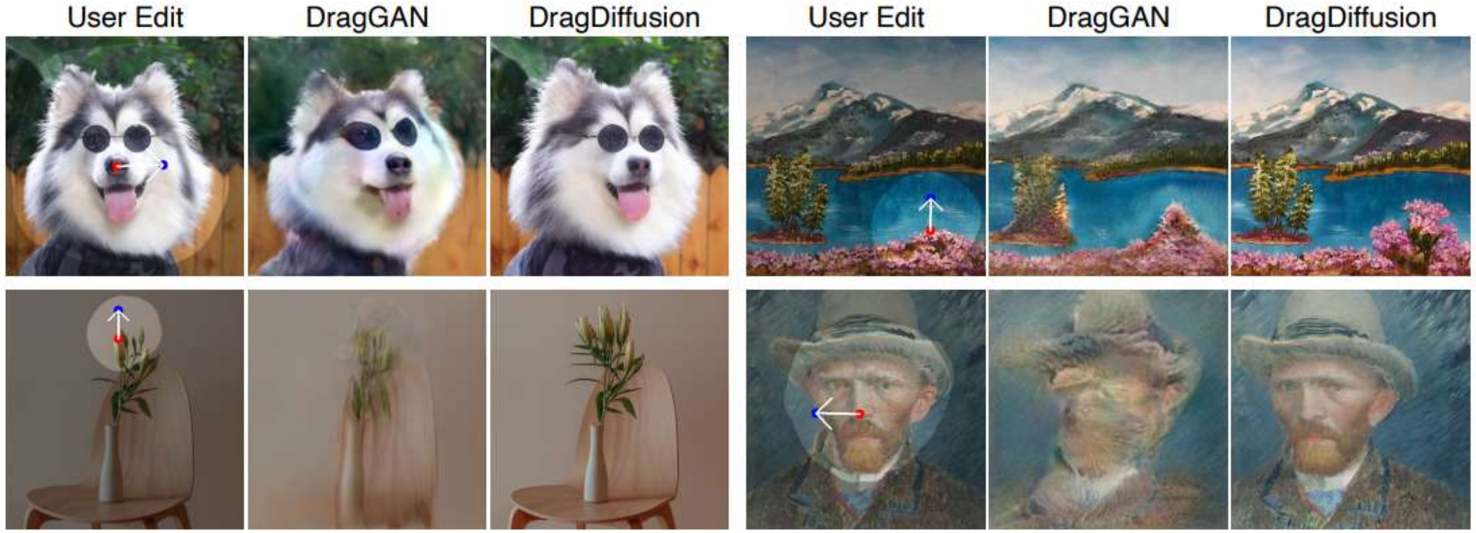
Target point



Existing approach

- DragGAN (2023)
  - Due to capacity of GAN model, generality is not satisfied in DragGAN

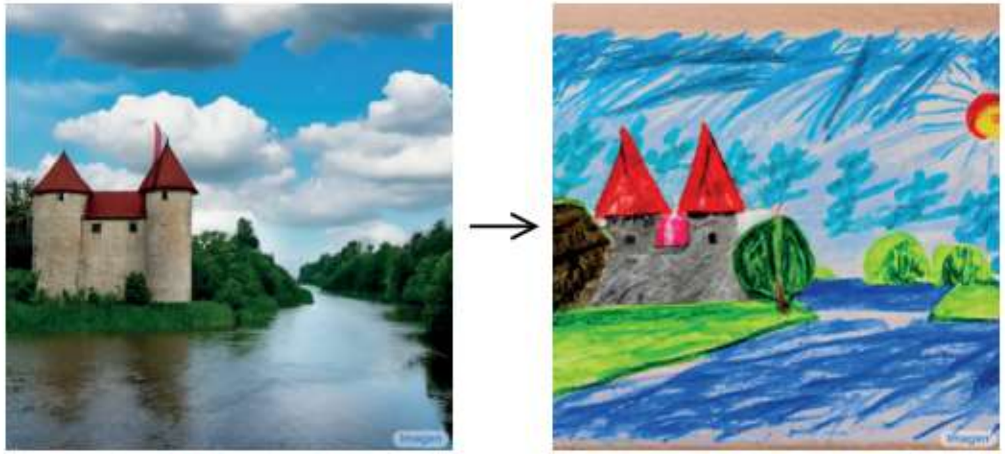
<DragGAN>



Existing approach

- Large-scale text-to-image diffusion models
  - Have a strong capabilities
  - Most diffusion-based editing models use text embeddings
  - It cannot achieve precise spatial control

InstructPix2Pix

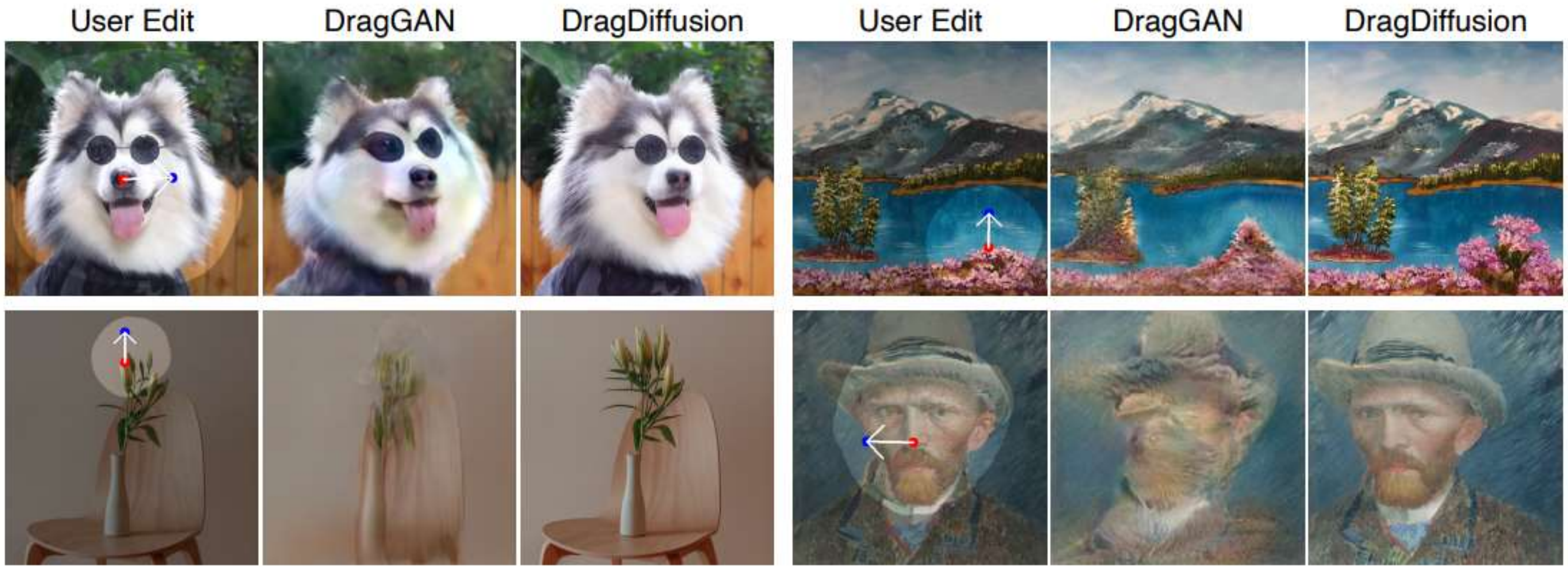


"Children drawing of a castle next to a river."

Prompt-to-Prompt

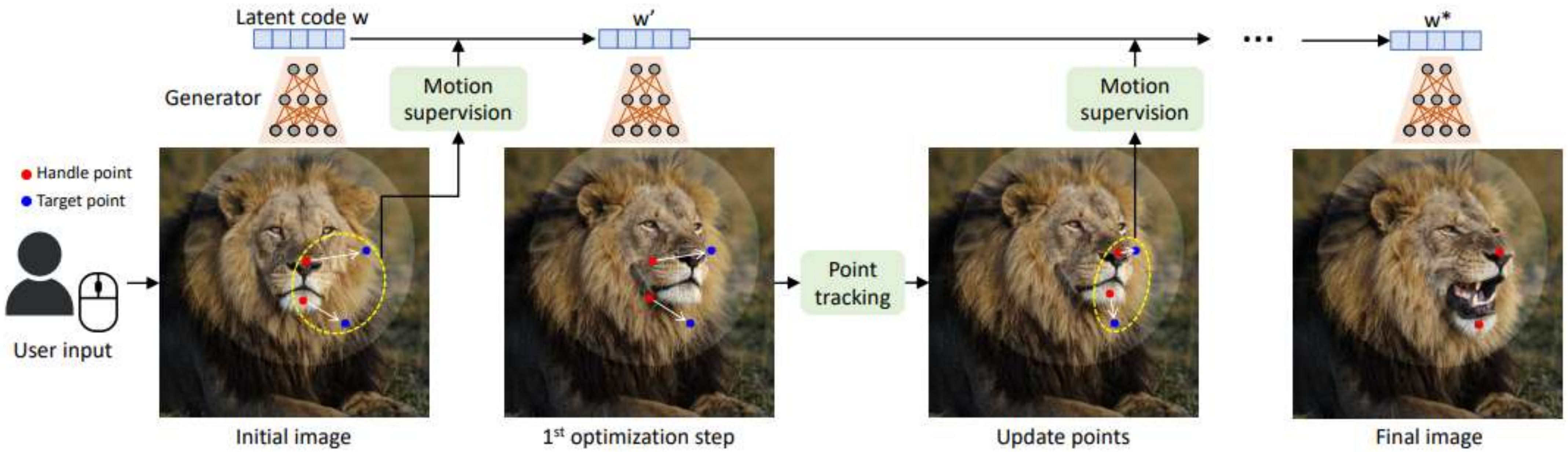
DragDiffusion

- Use diffusion models instead of GAN
- Additionally, introduce two techniques to preserve identity of original image
  - Identity-preserving fine-tuning
  - Reference-latent-control



Background

- DragGAN: image manipulation via optimizing latent code
  - StyleGAN2, feature map of 6th block
  - Motion supervision loss: move handle point to target point
  - Point tracking loss: update previous handle point to current handle point



Background

- DragGAN: image manipulation via optimizing latent code
  - Motion supervision loss: move handle point to target point
  - Point tracking loss: update previous handle point to current handle point

$$\mathcal{L} = \sum_{i=0}^{n} \sum_{\boldsymbol{q}_i \in \Omega_1(\boldsymbol{p}_i, r_1)} \|\mathbf{F}(\boldsymbol{q}_i) - \mathbf{F}(\boldsymbol{q}_i + \boldsymbol{d}_i)\|_1 + \lambda \|(\mathbf{F} - \mathbf{F}_0) \cdot (1 - \mathbf{M})\|_1,$$

Motion supervision

$$\boldsymbol{p}_i := \underset{\boldsymbol{q}_i \in \Omega_2(\boldsymbol{p}_i, r_2)}{\arg\min} \|\mathbf{F}'(\boldsymbol{q}_i) - \boldsymbol{f}_i\|_1.$$

Point tracking

$F: feature\ map$

$F_0: initial\ feature\ map$

$i: number\ of\ points$

$d_i: unit\ vector\ towards\ target\ points$

$q_i: small\ patch\ around\ p_i$

$f_i: feature\ of\ initial\ handle\ point$

$F'(q_i): updated\ feature\ map\ at\ q_i$

Background

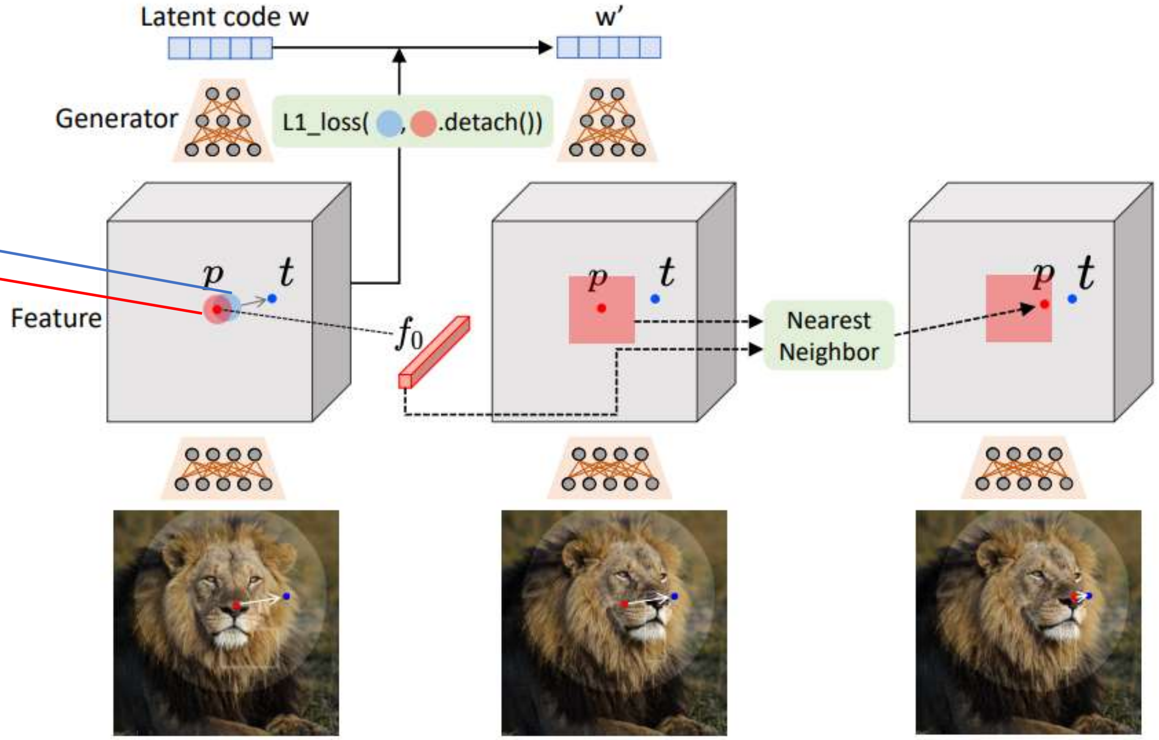
- DragGAN: image manipulation via optimizing latent code
  - Motion supervision loss: move handle point to target point
  - Point tracking loss: update previous handle point to current handle point

$$\mathcal{L} = \sum_{i=0}^{n} \sum_{\boldsymbol{q}_i \in \Omega_1(\boldsymbol{p}_i, r_1)} \|\mathbf{F}(\boldsymbol{q}_i) - \mathbf{F}(\boldsymbol{q}_i + \boldsymbol{d}_i)\|_1 + \lambda \|(\mathbf{F} - \mathbf{F}_0) \cdot (1 - \mathbf{M})\|_1,$$

Motion supervision

$$\boldsymbol{p}_i := \mathop{\arg\min}_{\boldsymbol{q}_i \in \Omega_2(\boldsymbol{p}_i, r_2)} \|\mathbf{F}'(\boldsymbol{q}_i) - f_i\|_1.$$

Point tracking



Background

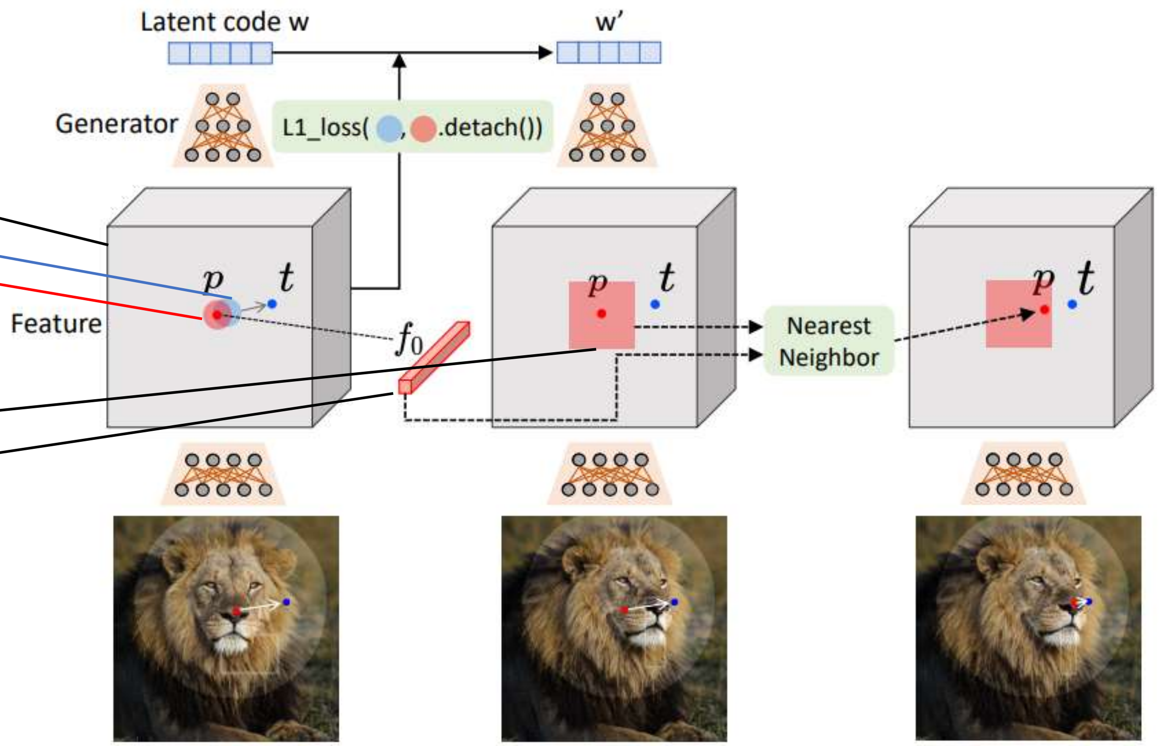
- DragGAN: image manipulation via optimizing latent code
  - Motion supervision loss: move handle point to target point
  - Point tracking loss: update previous handle point to current handle point

$$\mathcal{L} = \sum_{i=0}^{n} \sum_{\boldsymbol{q}_i \in \Omega_1(\boldsymbol{p}_i, r_1)} \|\mathbf{F}(\boldsymbol{q}_i) - \mathbf{F}(\boldsymbol{q}_i + \boldsymbol{d}_i)\|_1 + \lambda \|(\mathbf{F} - \mathbf{F}_0) \cdot (1 - \mathbf{M})\|_1,$$

Motion supervision

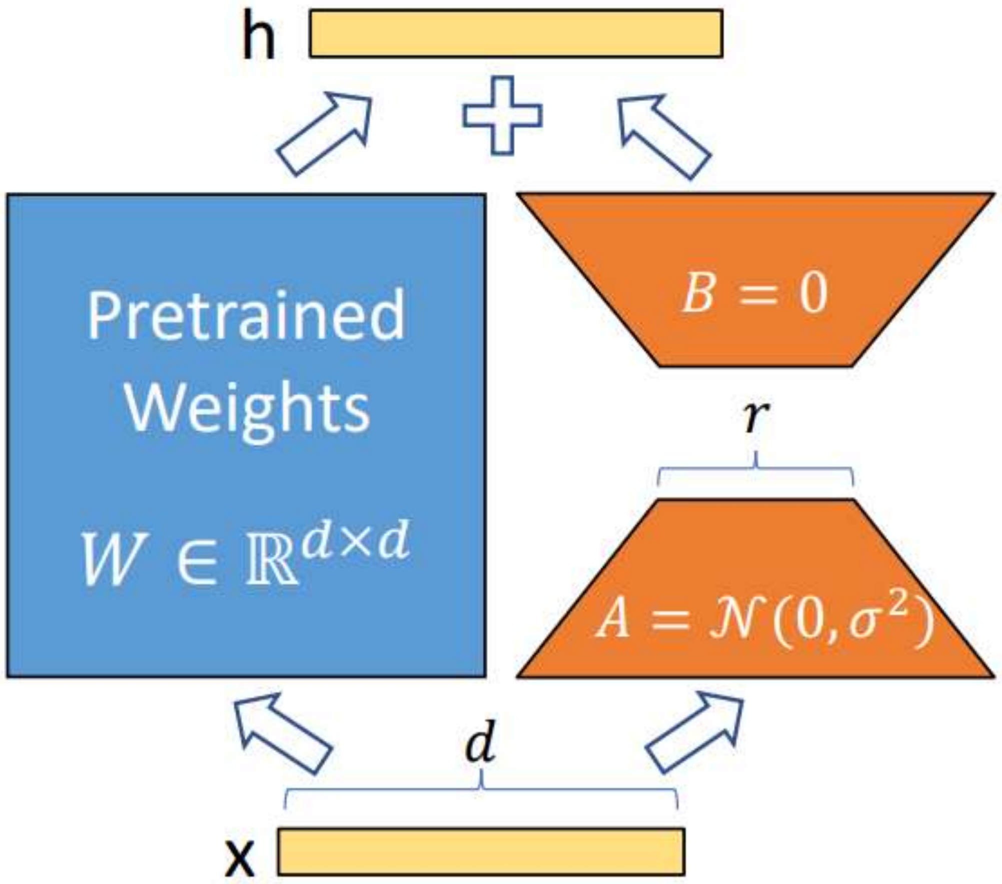
$$\boldsymbol{p}_i := \underset{\boldsymbol{q}_i \in \Omega_2(\boldsymbol{p}_i, r_2)}{\arg\min} \|\mathbf{F}'(\boldsymbol{q}_i) - f_i\|_1.$$

Point tracking



Background

- LoRA
  - Train the model with additional parameter



Background

- Diffusion models
  - Forward process: add noise / Reverse process: remove noise
  - Given data X_0, add noise $\epsilon \sim N(0, I)$ iteratively (*forward process*)
  - The model trained the forward process in reverse (*reverse process*)
  - Therefore, the model predicts the noise at a specific time step

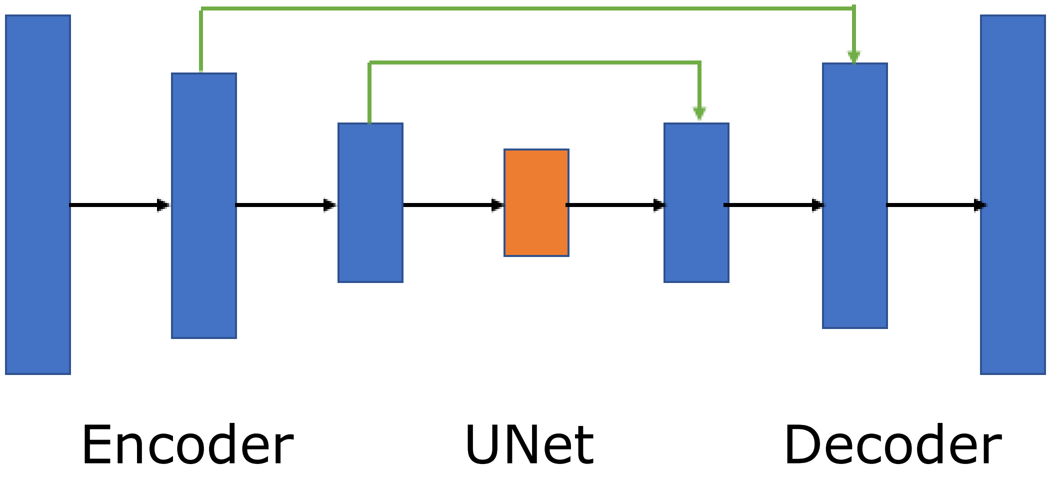
$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{(1 - \bar{\alpha}_t)}\epsilon$$

Forward process (add noise)

Reverse process (remove noise)

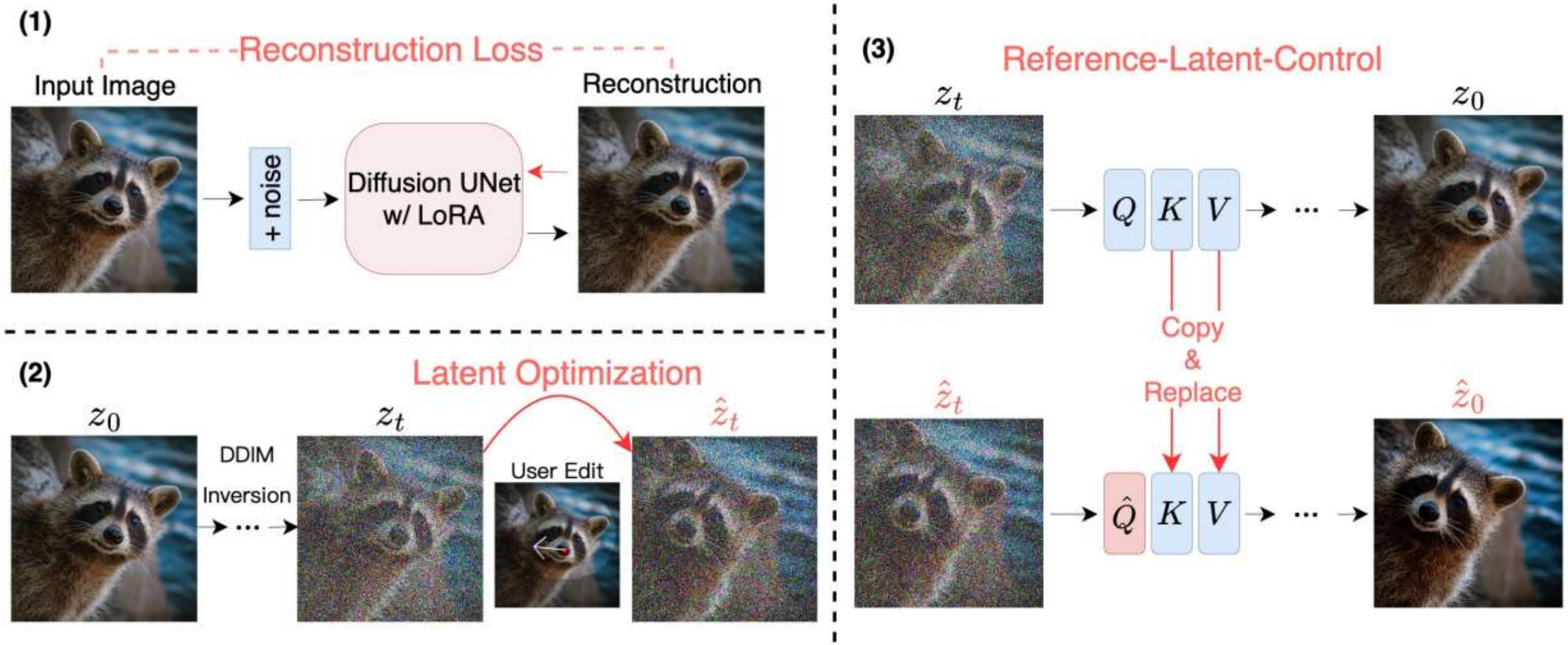
Background

- Diffusion models
  - Forward process: add noise / Reverse process: remove noise
  - Given data X_0, add noise $\epsilon \sim N(0, I)$ iteratively (*forward process*)
  - The model trained the forward process in reverse (*reverse process*)
  - Therefore, the model predicts the noise at a specific time step



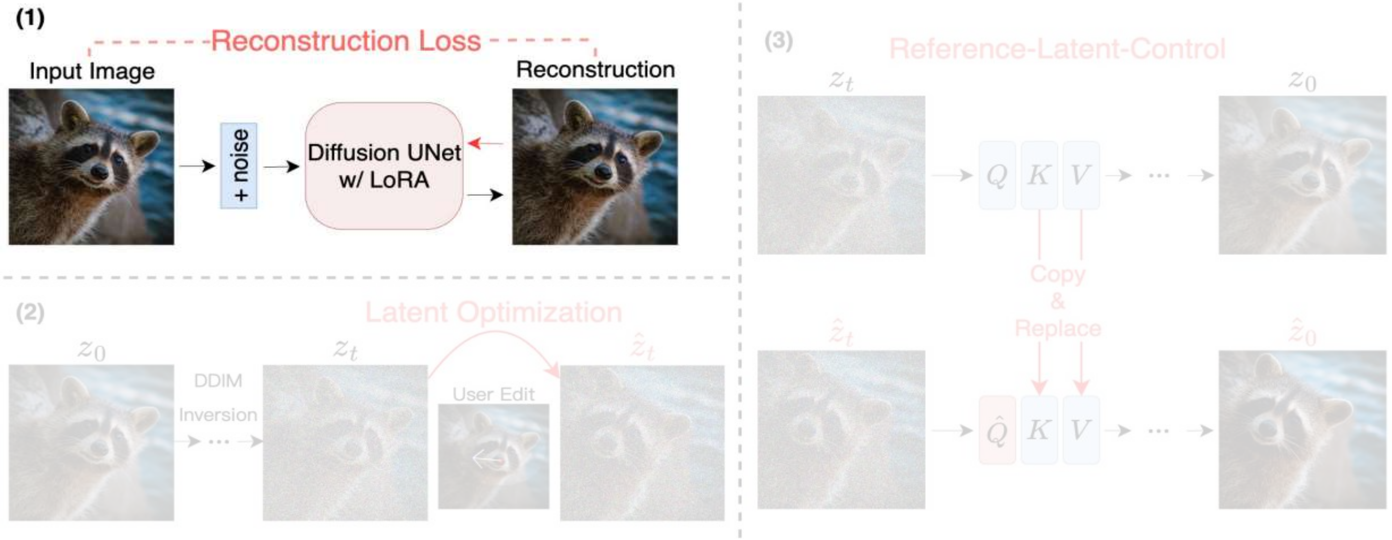
Method

- DragDiffusion
  - Use diffusion model
  - It consists of 3 step



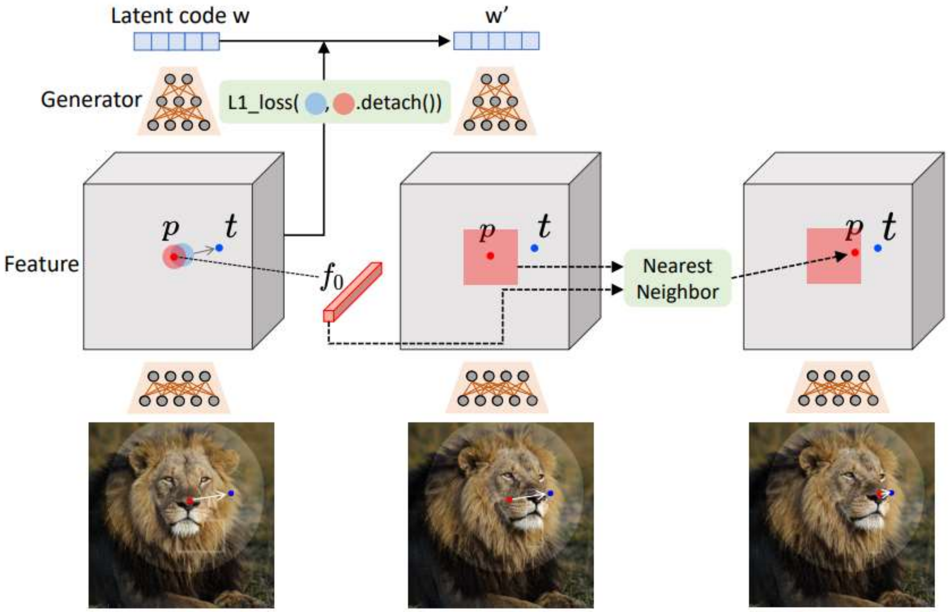
Method

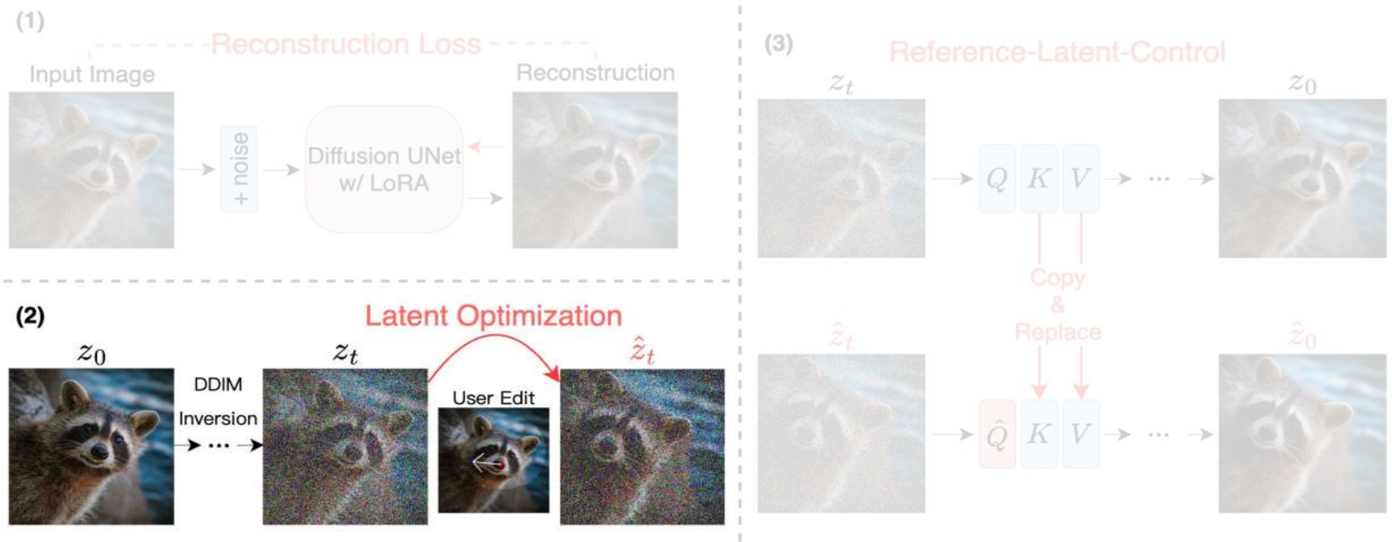
1. Identity-preserving fine-tuning
   - To encode input's feature in diffusion model, train model using LoRA
   - Only 80 steps
     - Subject-driven image generation: require 1000 steps (DreamBooth, textual inversion)
     - Sampling time: GAN < Diffusion + fine-tuning
     - A100GPU: 25 seconds



Method

2. Diffusion latent optimization (motion supervision + point tracking)
   - Move handle point to target point + update handle point
   - Same with DragGAN





Method

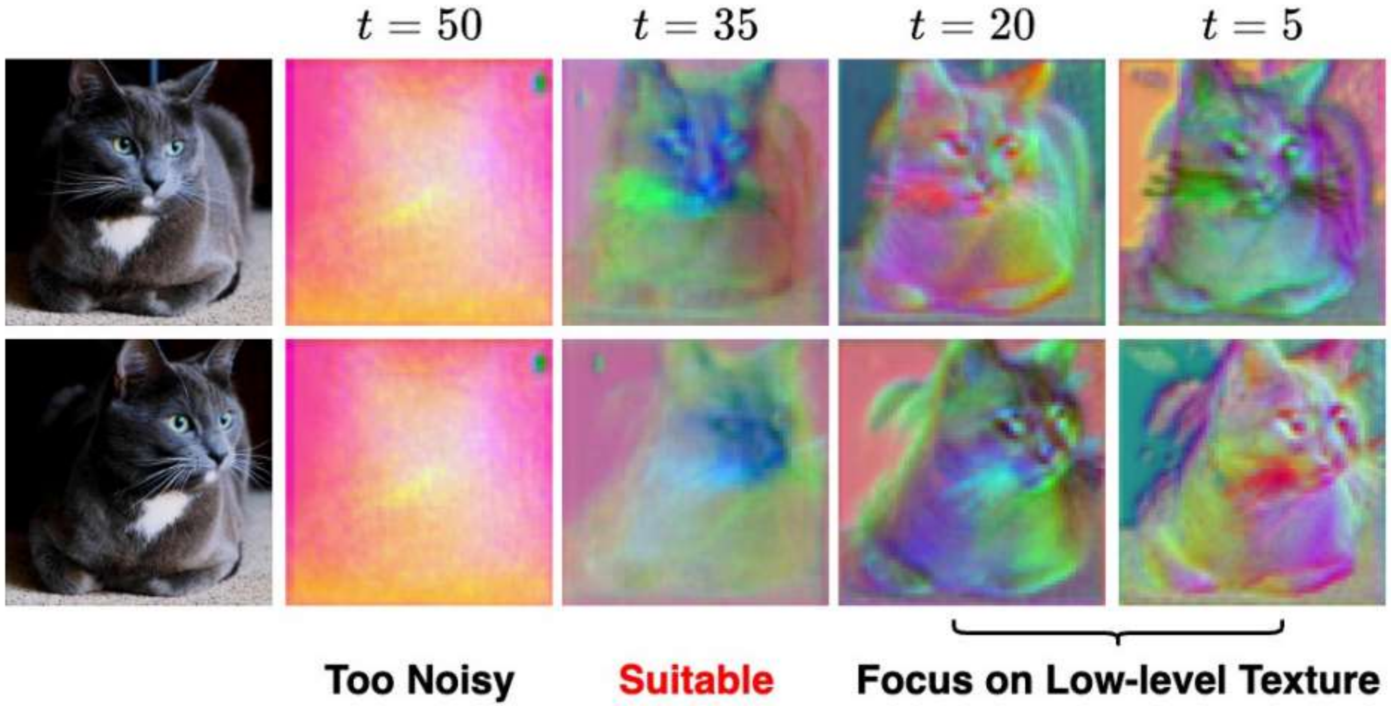
2. Diffusion latent optimization (motion supervision + point tracking)

- GAN: generate an image at once
- Diffusion: generate an image with iterative denoising

All of time steps? Certain time step?

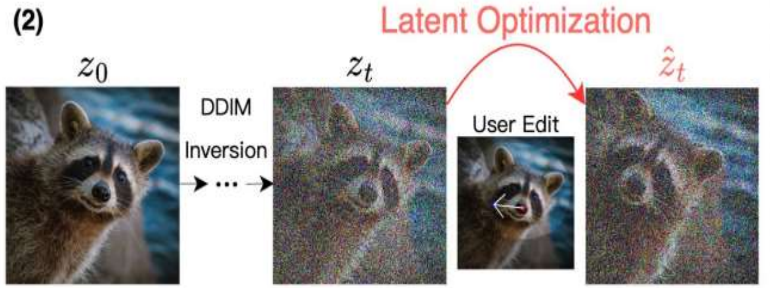
Method

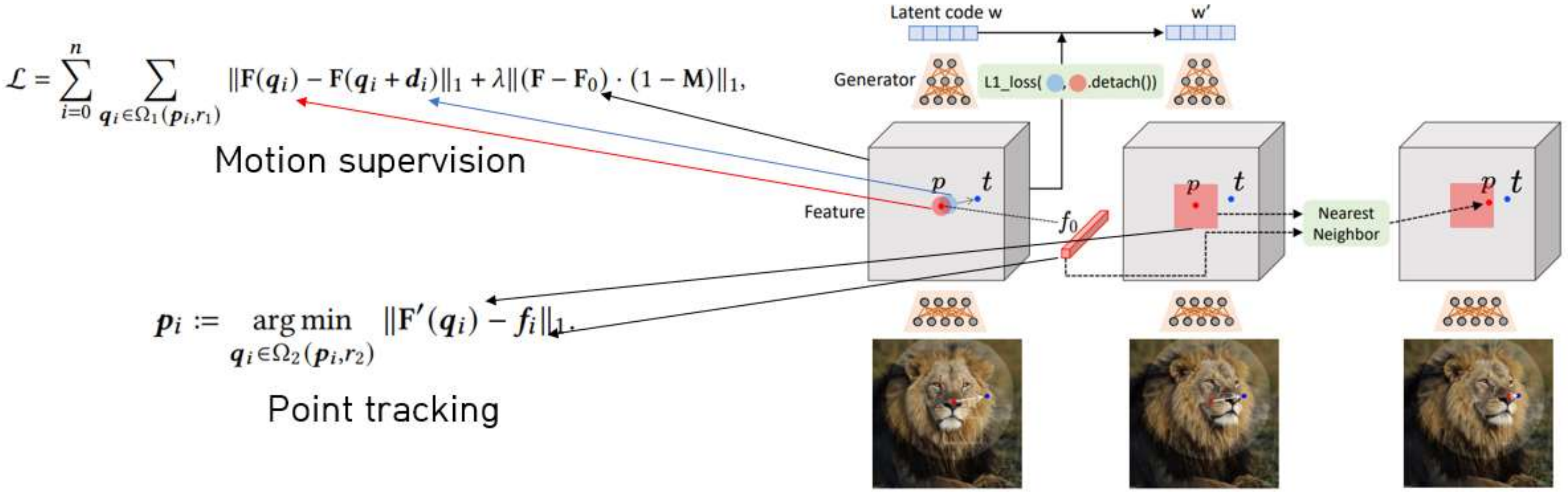
2. Diffusion latent optimization (motion supervision + point tracking)
   - Given two frames, visualize feature map over time using PCA
   - At $t = 35$, it has sufficient semantic and geometric information (shape, pose, etc)
   - Conduct optimization at certain time step ($t = 35$)



Method

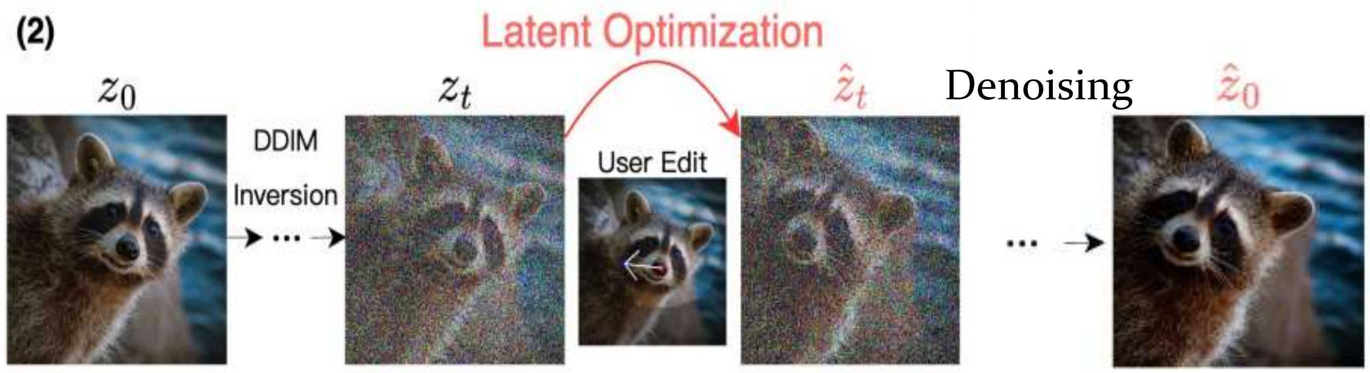
2. Diffusion latent optimization (motion supervision + point tracking)
   - Optimization step: 80





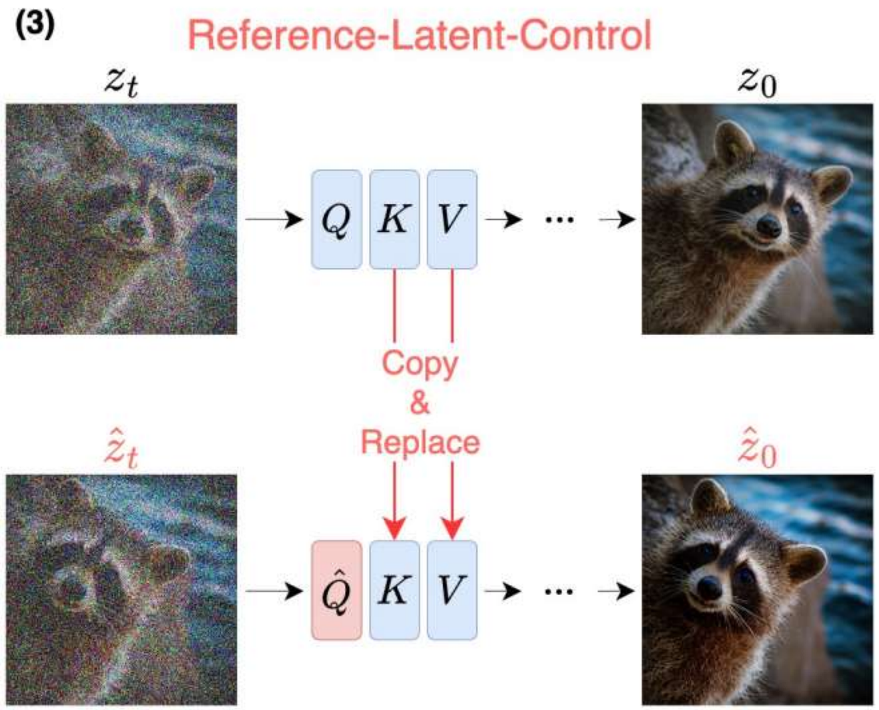
Method

3. Reference-latent-control
   - After optimization, denoise optimized latents to generate final editing results
   - Occurs defects: shift, degrade quality
   - Assume that this issue arises due to the absence of proper guidance from the original image during denoising process



Method

3. Reference-latent-control
   - In self-attention module, replace key and value of optimized latents with key and value of original latents
   - Improve consistency by referencing the correlated contents and texture of original image

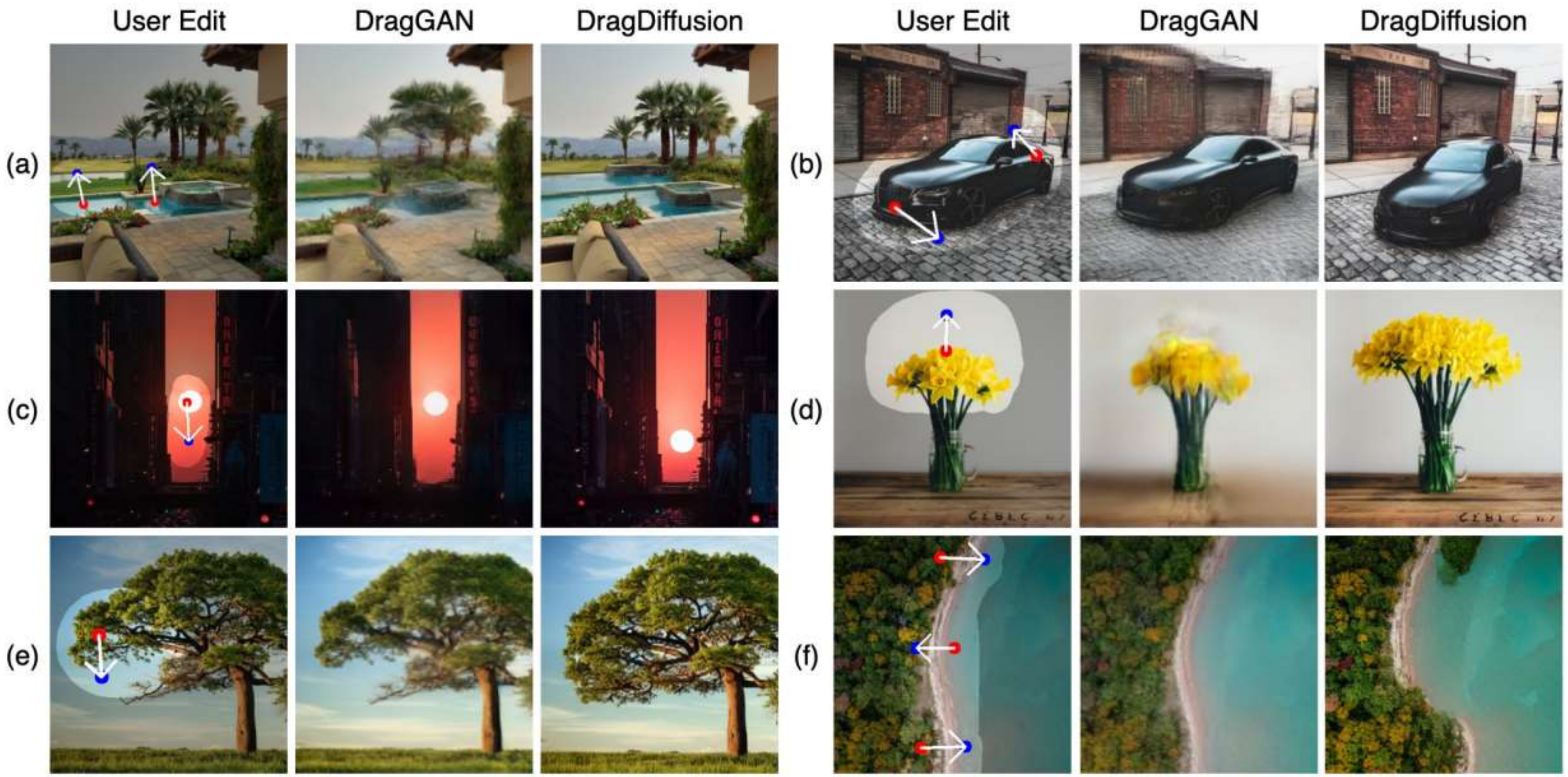


Experiments

- Models: Stable Diffusion v1.5
- LoRA fine-tuning: 80 steps
- Evaluation
  - Image Fidelity ↑ (IF): quantifies the similarity between original and edited images
  - Mean Distance ↓ (MD): how well the approach moves the semantic contents to the target points

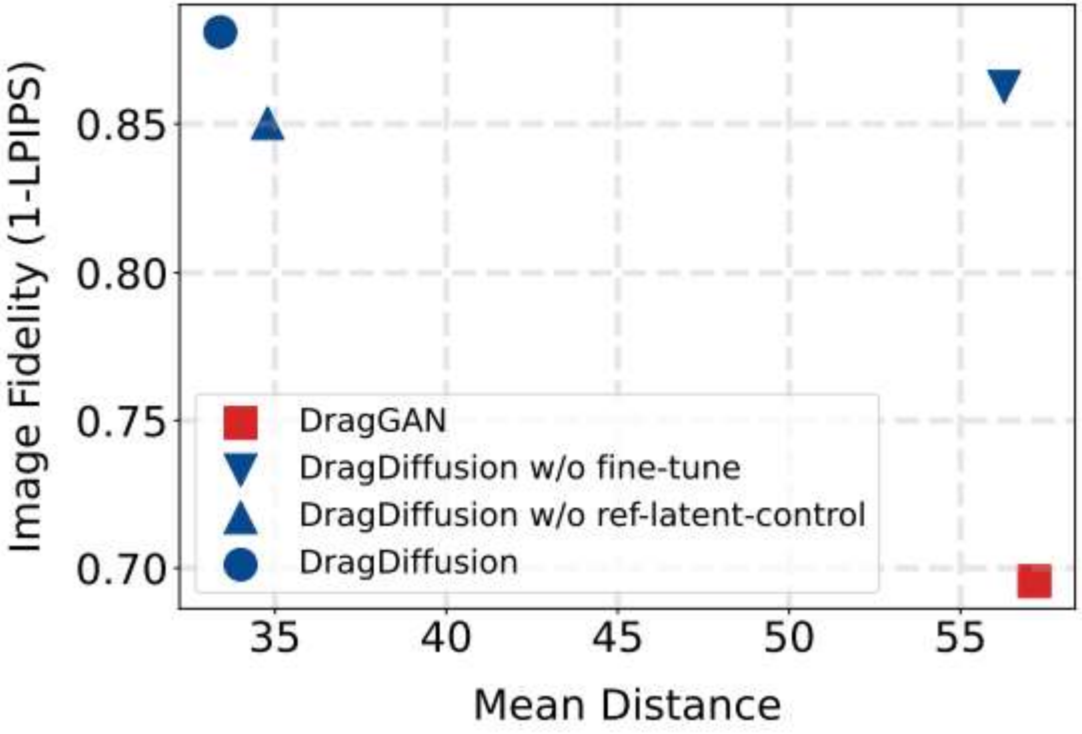
Experiments

- All results are obtained under the same user edit
  - Measure the generality between GAN and Diffusion



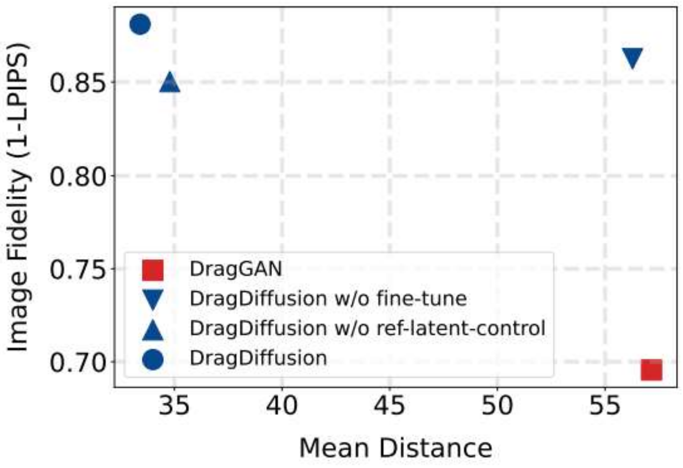
Experiments

- Comparison with DragGAN
  - Image Fidelity ↑ (IF): quantifies the similarity between original and edited images
  - Mean Distance ↓ (MD): how well the approach moves the semantic contents to the target points



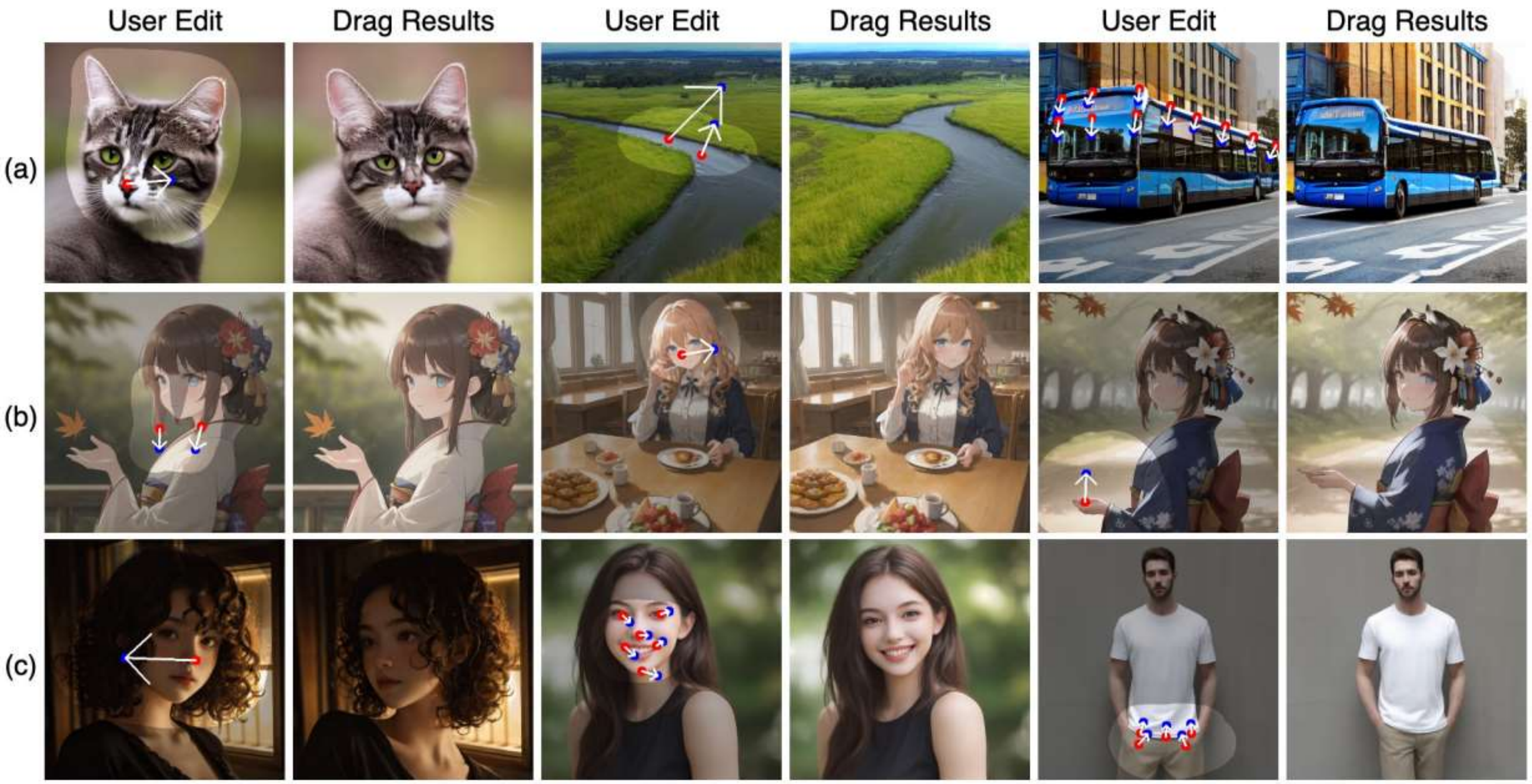
Experiments

- Ablation study
  - Image Fidelity ↑ (IF): quantifies the similarity between original and edited images
  - Mean Distance ↓ (MD): how to the target points



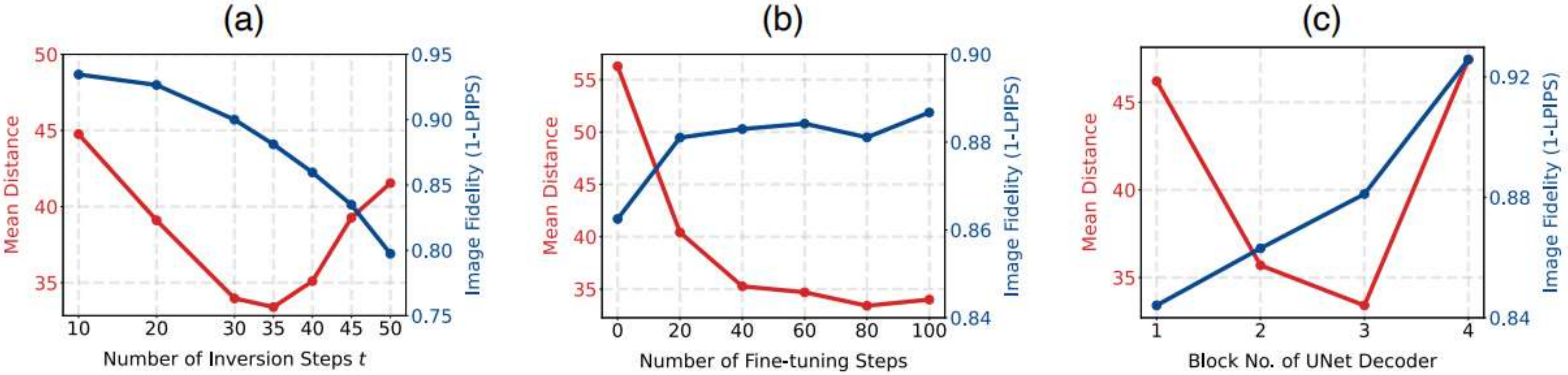
Experiments

- Show the generality of DragDiffusion



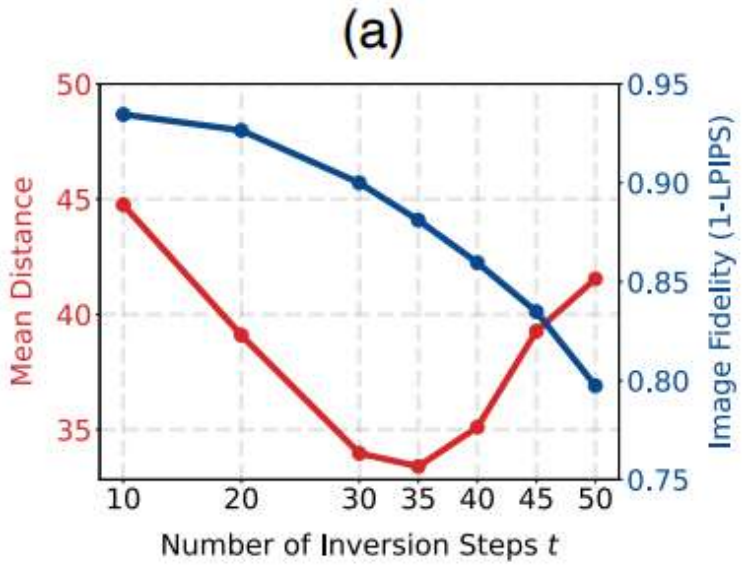
Experiments

- Ablation study
  - (a): $t = 35$
  - (b): LoRA 80 steps
  - (c): which is better results to apply optimization loss



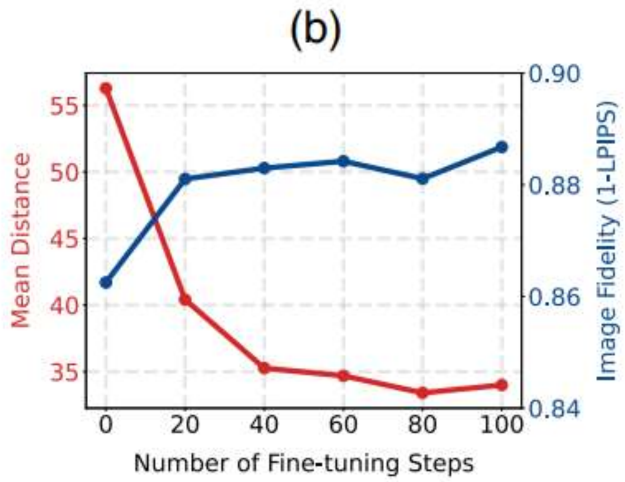
Experiments

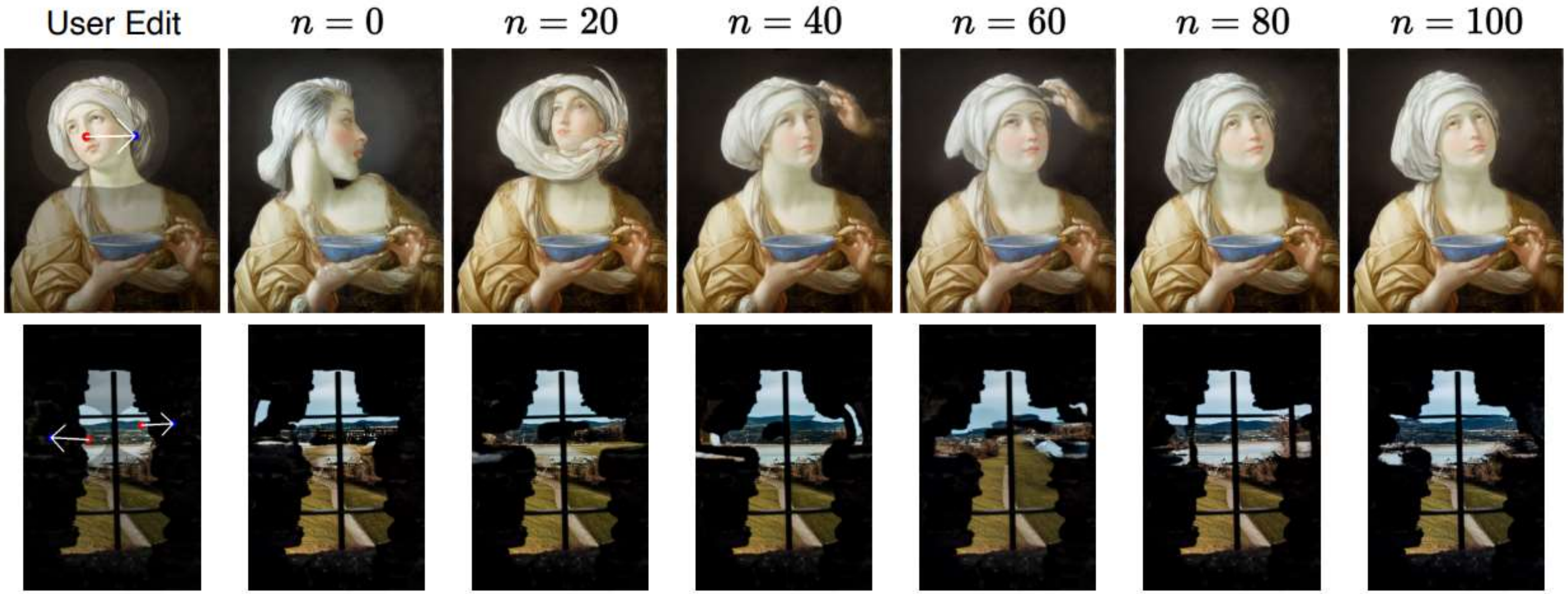
- Ablation study
  - Better results in $t \in [30,40]$



Experiments

- Ablation study
  - $n \geq 80$: produce reasonable results without artifacts





Experiments

- Ablation study
  - Conduct 3rd decoder block in UNet
  - 1st, 2nd: precise spatial control x
  - 4th: insufficient semantic and geometric information