Drag Your GAN

Point-based Manipulation on the Generative Image Manifold

2023

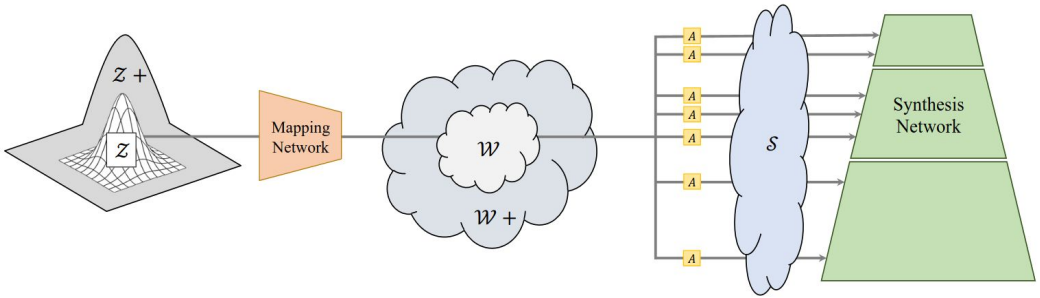
- Берем нашу картинку
- Ставим начальную и конечную точки (можно много точек)
- Редактирование

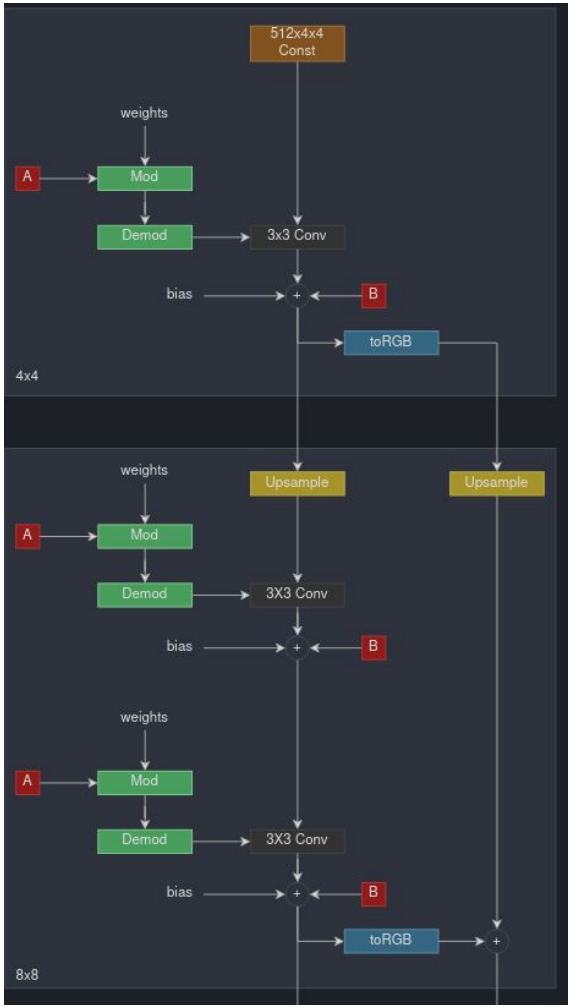

Network & latent
Pickle
stylegan2_lions_512_pytorch.pkl
Recent...
Browse...
Latent
7
Seed
0.001
Step Size
Reset
w
w+
Drag
Drag
Add point
Reset point
Start
Stop
Steps 0
Mask
Flexible area
Fixed area
Reset mask
Show mask
112
Radius
20
Lambda
Capture
Capture
_screenshots
Save image
Save GUI

План доклада

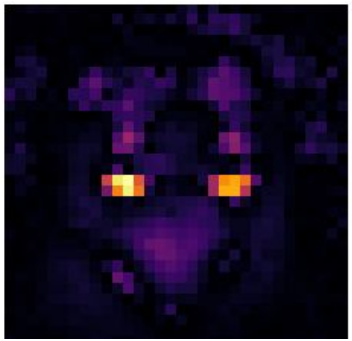
- Повторение StyleGAN2
- Пайплайн метода
- Сравнение с другими методами

StyleGAN2

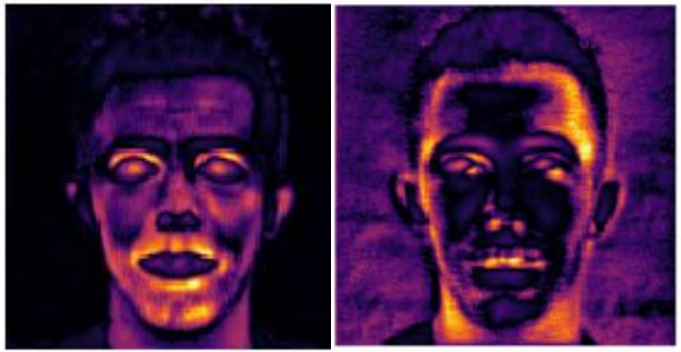

z +
z
Mapping Network
w
w +
A
A
A
A
A
A
s
Synthesis Network


512x4x4 Const
weights
A
Mod
Demod
3x3 Conv
bias
B
toRGB
4x4
weights
Upsample
Upsample
A
Mod
Demod
3X3 Conv
bias
B
weights
A
Mod
Demod
3X3 Conv
bias
B
toRGB
8x8

StyleGAN2 Feature Maps



16*16



128*128

Pipeline

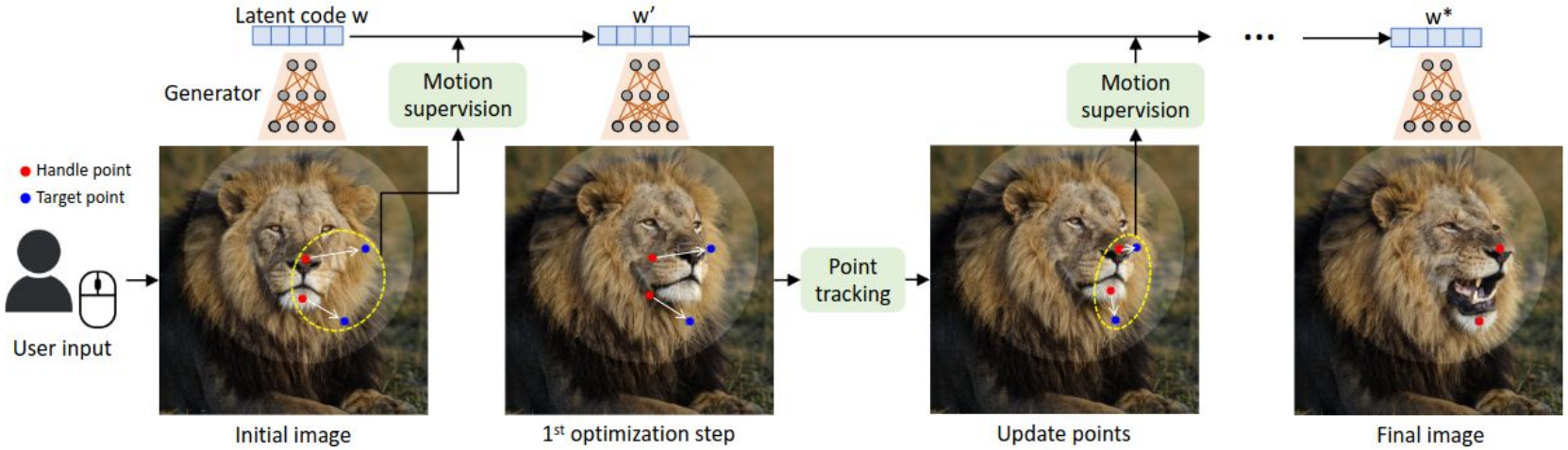


Fig. 2. Overview of our pipeline. Given a GAN-generated image, the user only needs to set several handle points (red dots), target points (blue dots), and optionally a mask denoting the movable region during editing (brighter area). Our approach iteratively performs *motion supervision* (Sec. 3.2) and *point tracking* (Sec. 3.3). The motion supervision step drives the handle points (red dots) to move towards the target points (blue dots) and the point tracking step updates the handle points to track the object in the image. This process continues until the handle points reach their corresponding target points.

30-200 iterations to converge

Motion supervision

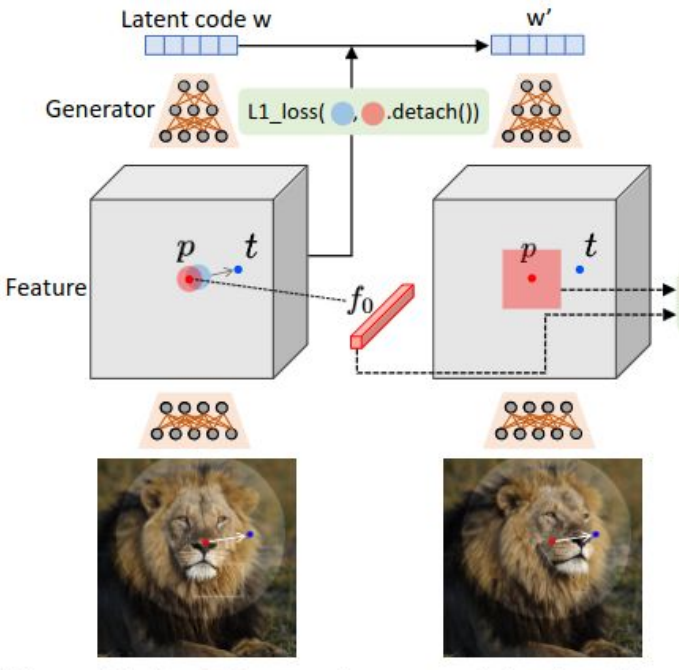


Image I

Image I'

We have

- Image **I** with corresponding latent code **w**
- Its intermediate feature maps **F0**
- Handle points p_i
- Target points t_i
- Binary mask **M** (optional)

After motion supervision:

- New image **I'** with corresponding **w'**

During motion supervision:

- r1 - radius in pixels (usually 3)
- $\Omega_1(p_i, r_1)$ - all points within our radius
- $d_i = \frac{t_i - p_i}{\|t_i - p_i\|_2}$ - normalized target direction

$$\mathcal{L} = \sum_{i=0}^{n} \sum_{q_i \in \Omega_1(p_i, r_1)} \|\mathbf{F}(q_i) - \mathbf{F}(q_i + d_i)\|_1 + \lambda \|(\mathbf{F} - \mathbf{F}_0) \cdot (1 - \mathbf{M})\|_1,$$

Loss is used to optimize **w** in latent space

Point Tracking

After motion supervision:

- New image **I'** with corresponding **w'**

Point Tracking

- Feature of the initial handle point $f_i = \mathbf{F}_0(\boldsymbol{p}_i)$.
- Patch around $\boldsymbol{p}_i$:

$$\Omega_2(\boldsymbol{p}_i, r_2) = \{(x, y) \mid |x - x_{p,i}| < r_2, |y - y_{p,i}| < r_2\}$$

Then the tracked point is obtained by searching for the nearest neighbor of f_i in $\Omega_2(\boldsymbol{p}_i, r_2)$:

$$\boldsymbol{p}_i := \underset{\boldsymbol{q}_i \in \Omega_2(\boldsymbol{p}_i, r_2)}{\arg\min} \|\mathbf{F}'(\boldsymbol{q}_i) - f_i\|_1. \quad (2)$$

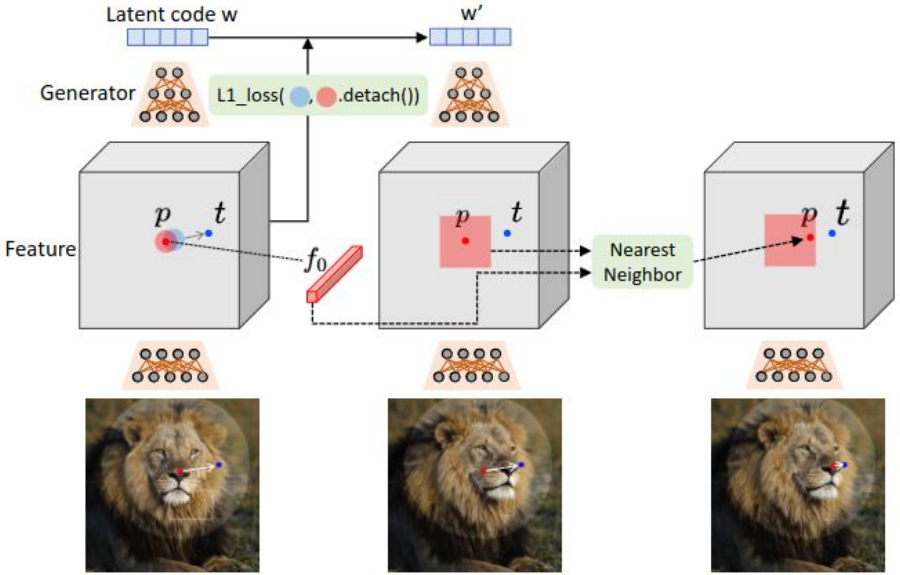


Fig. 3. Method. Our motion supervision is achieved via a shifted patch loss on the feature maps of the generator. We perform point tracking on the same feature space via the nearest neighbor search.

Comparison to UserContolableLT

Fig. 4. Qualitative comparison of our approach to UserControllableLT [Endo 2022] on the task of moving handle points (red dots) to target points (blue dots). Our approach achieves more natural and superior results on various datasets. More examples are provided in Fig. 10.

Face Landmark Manipulation

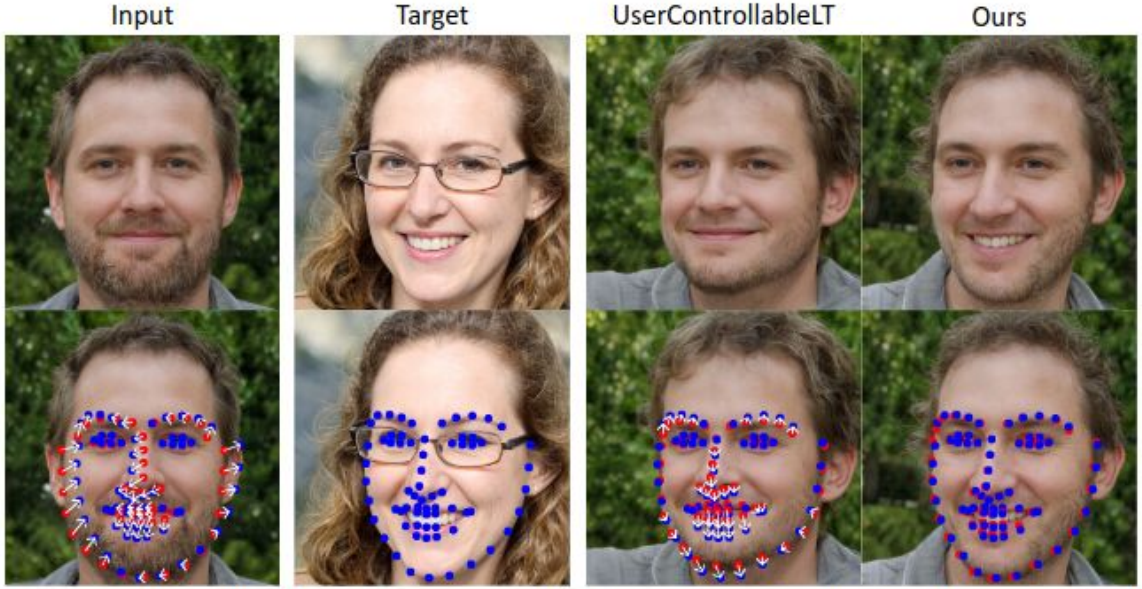


Fig. 7. Face landmark manipulation. Compared to UserControllableLT [Endo 2022], our method can manipulate the landmarks detected from the input image to match the landmarks detected from the target image with less matching error.

Adding Mask

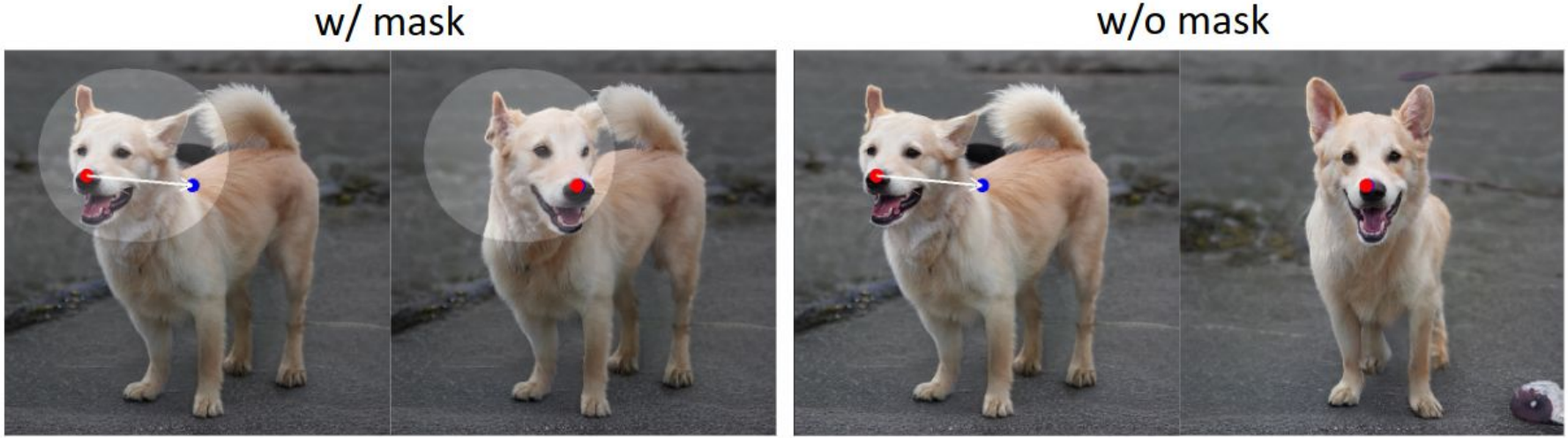


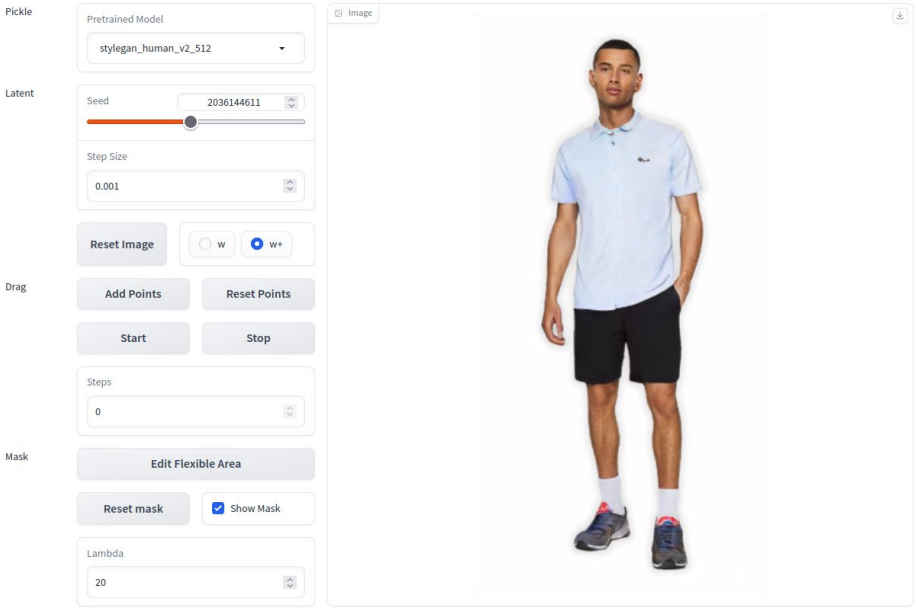
Fig. 8. Effects of the mask. Our approach allows masking the movable region. After masking the head region of the dog, the rest part would be almost unchanged.

Out-of-distribution manipulations

Fig. 9. Out-of-distribution manipulations. Our approach has extrapolation capability for creating images out of the training image distribution, for example, an extremely opened mouth and a greatly enlarged wheel.

Links

- https://vcai.mpi-inf.mpg.de/projects/DragGAN/data/paper.pdf
- https://huggingface.co/spaces/DragGan/DragGan
- https://github.com/OpenGVLab/DragGAN



Bilinear Interpolation