CLIP-NeRF:

Text-and-Image Driven Manipulation of Neural Radiance Fields

Can Wang
City University of Hong Kong
cwang355-c@my.cityu.edu.hk

Menglei Chai
Snap Inc.
cmlatsim@gmail.com

Mingming He
USC Institute for Creative Technologies
hmm.lillian@gmail.com

Dongdong Chen
Microsoft Cloud AI
cddlyf@gmail.com

Jing Liao*
City University of Hong Kong
jingliao@cityu.edu.hk

target

- 对于NeRF的condition-driven manipulation,主要是改变shape和appearance

问题

- 形状和颜色相互联系，另外一个会随着其中一个改变
- 想要解耦appearance 和 shape

Method

$$F : (\mathbf{x}, \mathbf{v}) \rightarrow (\mathbf{c}, \sigma)$$
$$F_\theta : (\mathbf{x}, \mathbf{v}, \mathbf{z_s}, \mathbf{z_a}) \rightarrow (\mathbf{c}, \sigma)$$
$$(\Gamma(\mathbf{x}) \oplus z_s, \Gamma(\mathbf{v}) \oplus z_a) \rightarrow (\mathbf{c}, \sigma)$$

Solution

Shape Deformation Network τ

$$\tau : (\mathbf{x}, z_s) \rightarrow \Delta\mathbf{x}$$
$$\Gamma^*(\mathbf{p}, z_s) = \{\gamma^*(p, \Delta p) | p \in \mathbf{p}, \Delta p \in \tau(\mathbf{p}, z_s)\}$$
$$\gamma^*(p, \Delta p)_k = \gamma(p)_k + tanh(\Delta p_k)$$

Deferred Appearance Conditioning

$$\mathcal{F}_\theta(\mathbf{x}, \mathbf{v}, z_s, z_a) : (\Gamma^*(\mathbf{x}, z_s), \Gamma(\mathbf{v}) \oplus z_a) \rightarrow (\mathbf{x}, \sigma)$$
$$\mathcal{F}_\theta(\mathbf{v}, z_s, z_a), x \in R$$

CLIP-Driven Manipulation

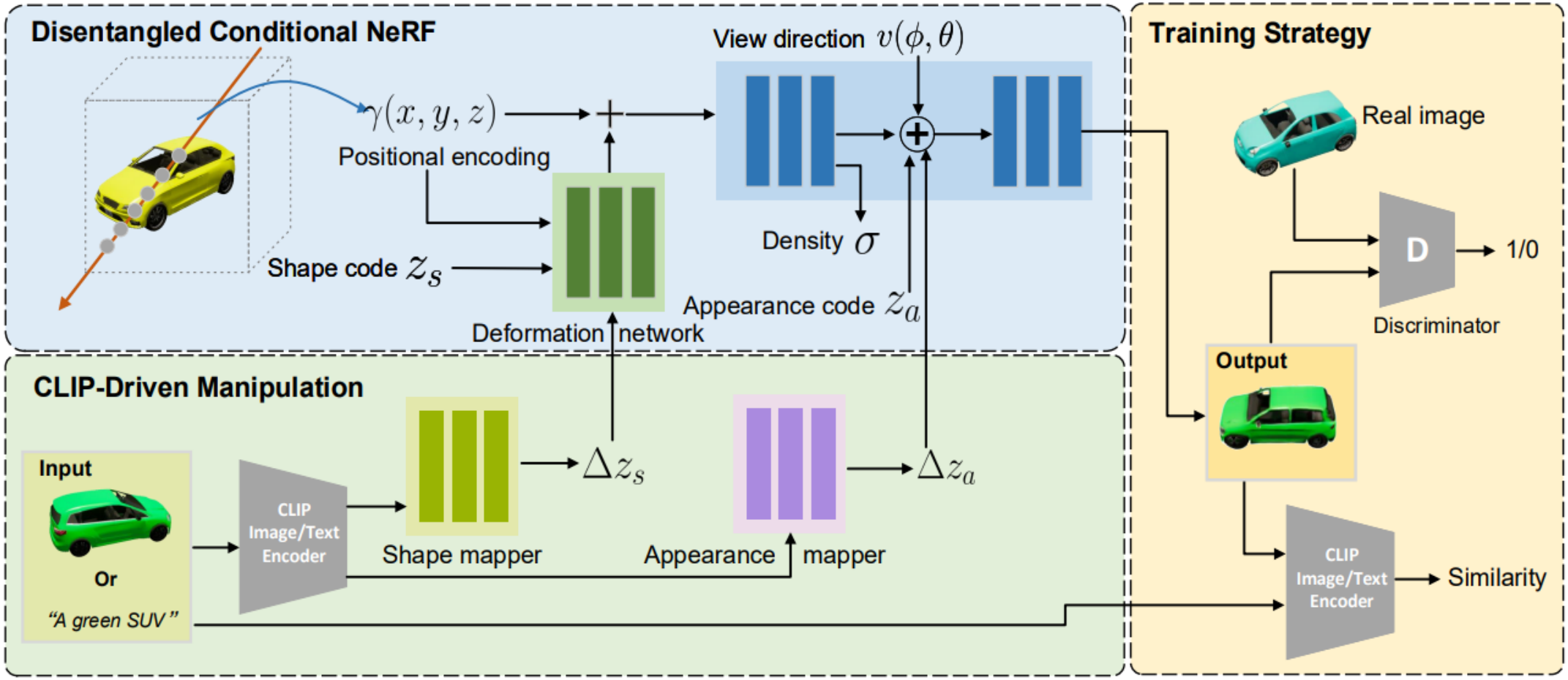
initial state: z_s', z_a'

trained Mapper: $\mathcal{M}_s, \mathcal{M}_a$

$$z_s = \mathcal{M}_s(\hat{\xi}_t(\mathbf{t})) + z_s',$$
$$z_a = \mathcal{M}_a(\hat{\xi}_t(\mathbf{t})) + z_a',$$

cross-modal CLIP distance function:

$$D_{CLIP}(\mathbf{I}, \mathbf{t}) = 1 - < \hat{\xi}_i(\mathbf{I}), \hat{\xi}_t(\mathbf{t}) >$$


Disentangled Conditional NeRF
View direction $v(\phi, \theta)$
$\gamma(x, y, z)$
Positional encoding
Density σ
Shape code z_s
Appearance code z_a
Deformation network
Training Strategy
Real image
D
1/0
Discriminator
Output
CLIP-Driven Manipulation
Input
Or
"A green SUV"
CLIP Image/Text Encoder
Shape mapper
Appearance mapper
Δz_s
Δz_a
CLIP Image/Text Encoder
Similarity

training strategy

two stages:

- disentangled conditional NeRF including the conditional NeRF generator($\mathcal{F}$) and the deformation network(τ);
- fix the weights of the generator($\mathcal{F}$) and train the CLIP manipulation parts including both the shape and appearance mappers.($\mathcal{M}_s, \mathcal{M}_a$)

Disentangled Conditional NeRF. Our conditional NeRF generator $\mathcal{F}_\theta$ is trained together with the deformation network using a non-saturating GAN objective [22] with the discriminator $\mathcal{D}$, where $f(x) = -\log\big(1+\exp(-x)\big)$ and λ_r is the regularization weight. Assuming that real images $\mathbf{I}$ form the training data distribution of d, we randomly sample the shape code $\boldsymbol{z}_s$, the appearance code $\boldsymbol{z}_a$, and the camera pose from $\mathcal{Z}_s$, $\mathcal{Z}_a$, and $\mathcal{Z}_v$, respectively, where $\mathcal{Z}_s$ and $\mathcal{Z}_a$ are the normal distribution, and $\mathcal{Z}_v$ is the upper hemisphere of the camera coordinate system.

$$\begin{aligned}\mathcal{L}_{\text{GAN}} =& \mathbb{E}_{\boldsymbol{z}_s\sim\mathcal{Z}_s,\boldsymbol{z}_a\sim\mathcal{Z}_a,\boldsymbol{v}\sim\mathcal{Z}_v}\Big[f\big(\mathcal{D}(\mathcal{F}_\theta(\boldsymbol{v},\boldsymbol{z}_s,\boldsymbol{z}_a))\big)\Big] \\ &+\mathbb{E}_{\mathbf{I}\sim d}\Big[f\big(-\mathcal{D}(\mathbf{I})+\lambda_r\|\nabla\mathcal{D}(\mathbf{I})\|^2\big)\Big].\end{aligned} \tag{7}$$

CLIP Manipulation Mappers. We use pre-trained NeRF generator $\mathcal{F}_\theta$, CLIP encoders $\{\hat{\mathcal{E}}_t, \hat{\mathcal{E}}_i\}$, and the discriminator $\mathcal{D}$ to train the CLIP shape mapper $\mathcal{M}_s$ and appearance mapper $\mathcal{M}_a$. All network weights, except the mappers, are fixed, denoted as $\{\hat{\cdot}\}$. Similar to the first stage, we randomly sample the shape code $\boldsymbol{z}_s$, the appearance code $\boldsymbol{z}_a$, and the camera pose $\boldsymbol{v}$ from their respective distributions. In addition, we sample the text prompt $\boldsymbol{t}$ from a pre-defined text library $\mathbf{T}$. By using our CLIP distance D_{CLIP} (Eq. 6) with weight λ_c, we train the mappers with the following losses:

$$\begin{aligned}\mathcal{L}_{\text{shape}} = f\big(\hat{\mathcal{D}}\big(\hat{\mathcal{F}}_\theta\big(\boldsymbol{v}, \mathcal{M}_s(\hat{\mathcal{E}}_t(\boldsymbol{t})) + \boldsymbol{z}_s, \boldsymbol{z}_a\big)\big)\big) + \\ \lambda_c D_{\text{CLIP}}\big(\hat{\mathcal{F}}_\theta\big(\boldsymbol{v}, \mathcal{M}_s(\hat{\mathcal{E}}_t(\boldsymbol{t})) + \boldsymbol{z}_s, \boldsymbol{z}_a\big), \boldsymbol{t}\big),\end{aligned} \tag{8}$$

$$\begin{aligned}\mathcal{L}_{\text{appear}} = f\big(\hat{\mathcal{D}}\big(\hat{\mathcal{F}}_\theta\big(\boldsymbol{v}, \boldsymbol{z}_s, \mathcal{M}_a(\hat{\mathcal{E}}_t(\boldsymbol{t})) + \boldsymbol{z}_a\big)\big)\big) + \\ \lambda_c D_{\text{CLIP}}\big(\hat{\mathcal{F}}_\theta\big(\boldsymbol{v}, \boldsymbol{z}_s, \mathcal{M}_a(\hat{\mathcal{E}}_t(\boldsymbol{t})) + \boldsymbol{z}_a\big), \boldsymbol{t}\big).\end{aligned} \tag{9}$$

Inverse Manipulation

> 获得初始参数，采用迭代优化法

To be specific, during each iteration, we first optimize $\boldsymbol{v}$ while keeping $\boldsymbol{z}_s$ and $\boldsymbol{z}_a$ fixed using the following loss:

$$\begin{aligned}\mathcal{L}_v = \left\|\hat{\mathcal{F}}_\theta(\boldsymbol{v}, \hat{\boldsymbol{z}}_s, \hat{\boldsymbol{z}}_a) - \mathbf{I}_r\right\|_2 + \\ \lambda_v D_{\text{CLIP}}\left(\hat{\mathcal{F}}_\theta(\boldsymbol{v}, \hat{\boldsymbol{z}}_s, \hat{\boldsymbol{z}}_a), \mathbf{I}_r\right).\end{aligned} \tag{10}$$

We then update the shape code by minimizing:

$$\mathcal{L}_s = \left\| \hat{\mathcal{F}}_\theta(\hat{\boldsymbol{v}}, \boldsymbol{z}_s + \lambda_n \boldsymbol{z}_n, \hat{\boldsymbol{z}_a}) - \mathbf{I}_r \right\|_2 + \lambda_s D_{\text{CLIP}}\left(\hat{\mathcal{F}}_\theta(\hat{\boldsymbol{v}}, \boldsymbol{z}_s + \lambda_n \boldsymbol{z}_n, \hat{\boldsymbol{z}_a}), \mathbf{I}_r\right),$$

The appearance code is updated in a similar manner:

$$\mathcal{L}_a = \left\| \hat{\mathcal{F}}_\theta(\hat{\boldsymbol{v}}, \hat{\boldsymbol{z}_s}, \boldsymbol{z}_a + \lambda_n \boldsymbol{z}_n) - \mathbf{I}_r \right\|_2 + \lambda_a D_{\text{CLIP}}\left(\hat{\mathcal{F}}_\theta(\hat{\boldsymbol{v}}, \hat{\boldsymbol{z}_s}, \boldsymbol{z}_a + \lambda_n \boldsymbol{z}_n), \mathbf{I}_r\right), \tag{12}$$

Experiments

datasets

- Photoshapes (chairs)
- Carla (cars)

comparsion

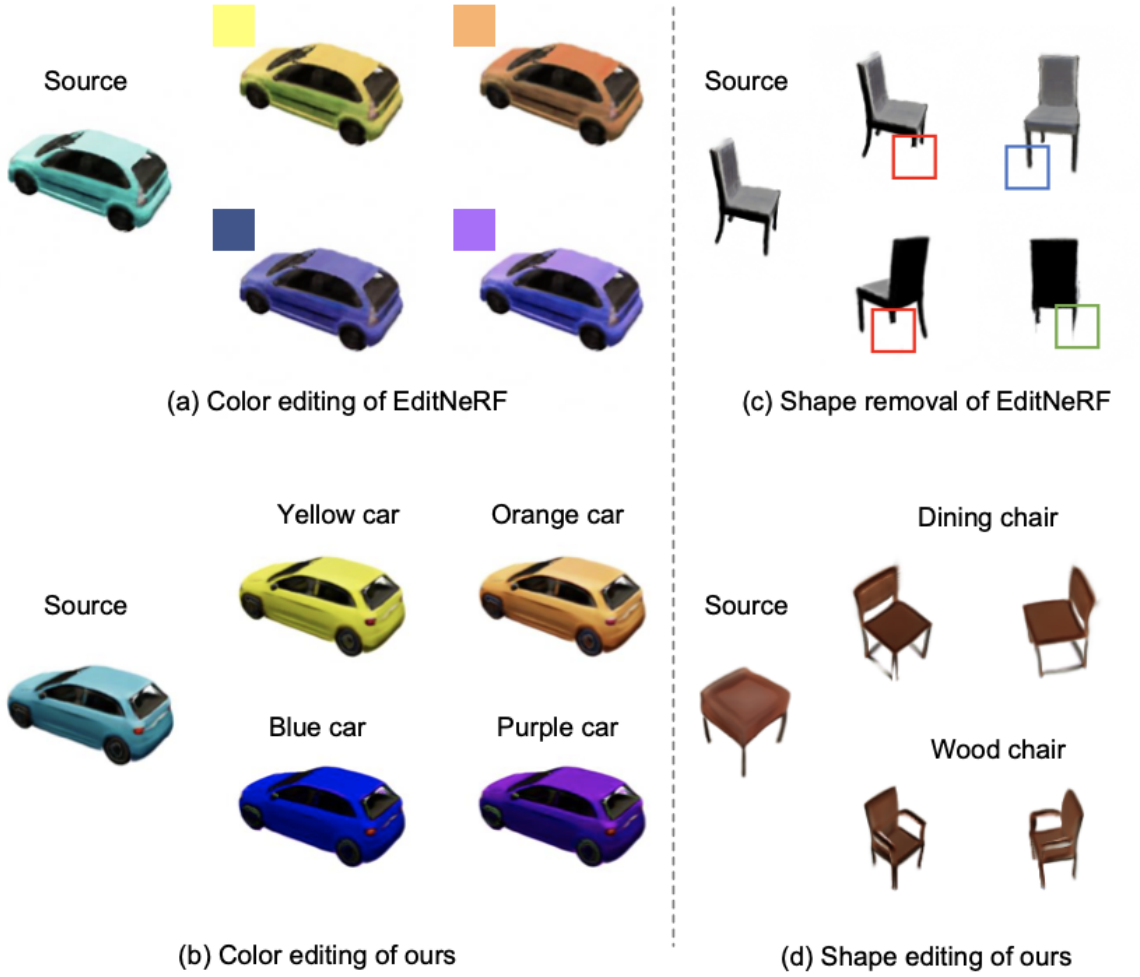


Figure 3. Compared to EditNeRF.

	Chairs		Cars	
	Shape	Appearance	Shape	appearance
EditNeRF	30.0	15.9	33.2	16.8
Ours	0.58	0.51	2.12	1.98

Table 1. **Compared to EditNeRF [21] on editing time averaged on 20 images.** We only include the inference/optimization time(s) and single-view rendered time(s) for chairs (128×128 pixels) and cars (256×256 pixels).

Ablation

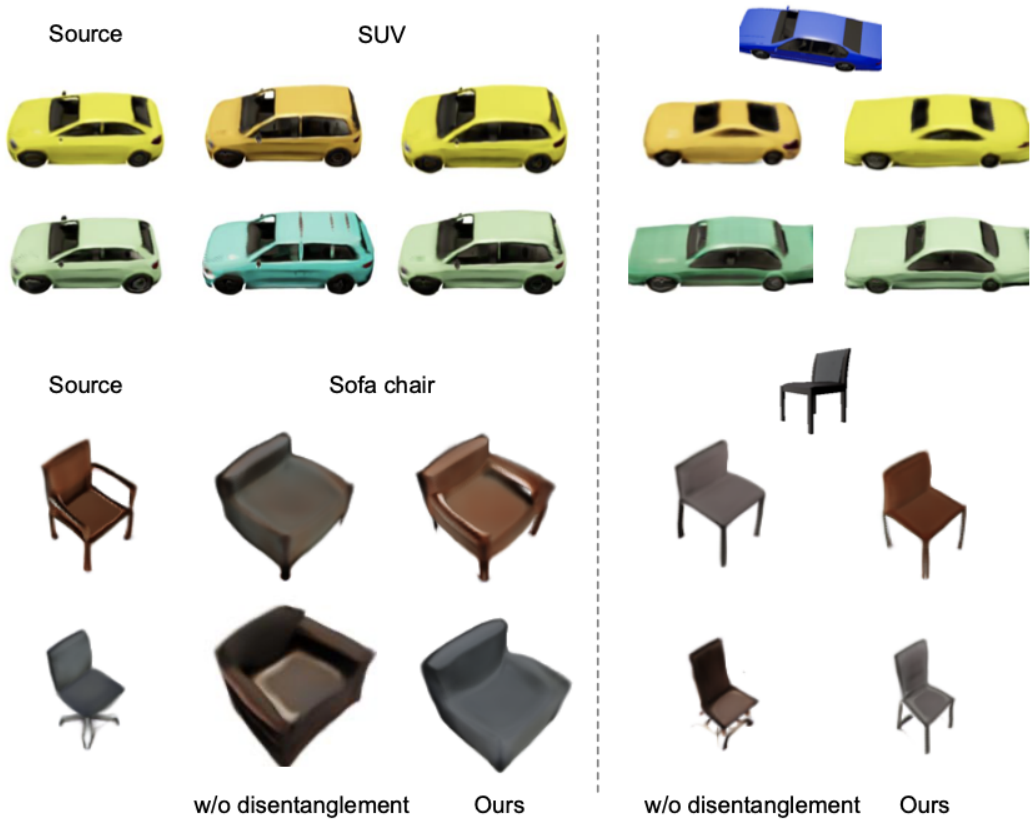


Figure 4. **Ablation Study for Disentanglement.** We show text-and-exemplar driven shape editing results of our method and the baseline method without using our disentangled technique. When editing the shape, the latter can change the appearance, while ours keeps the appearance unchanged.

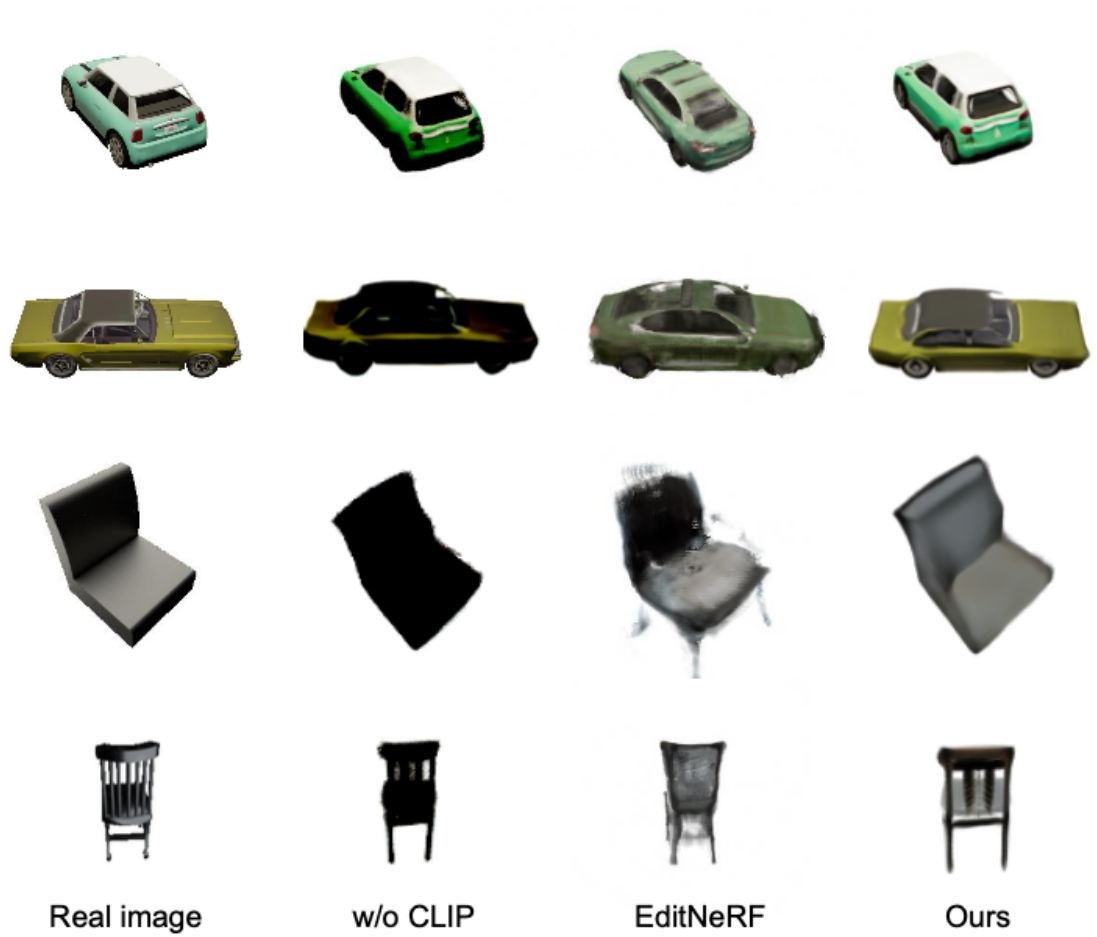


Figure 5. **Ablation study on our inversion method and comparison with EditNeRF.**

		Chairs			Cars		
		Before	After	Diff.	Before	After	Diff.
(a)	EditNeRF	36.8	40.2	3.4	102.8	118.7	15.9
	w/o disen.	52.5	54.3	1.8	69.2	69.9	0.7
	Ours	47.8	49.0	1.2	66.7	67.2	0.5
(b)	w/o disen.	52.5	53.2	0.7	69.2	71.1	1.9
	Ours	47.8	48.4	0.6	66.7	67.8	1.1

Table 2. **Fréchet inception distance (FID) for evaluating the image quality of reconstructed views before and after editing on: (a) color and (b) shape (lower value means better).** We use 2K

Results

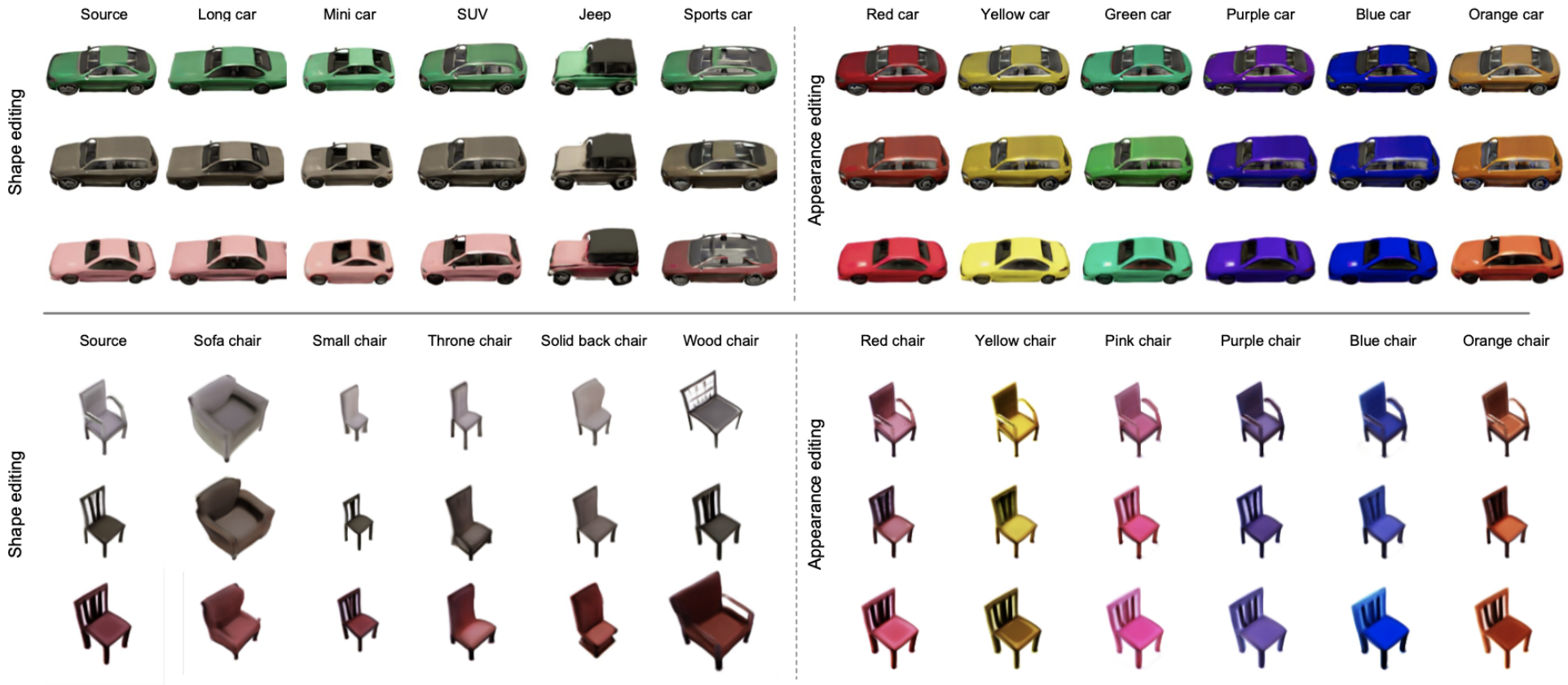


Figure 6. **Text-Driven Editing Results.**

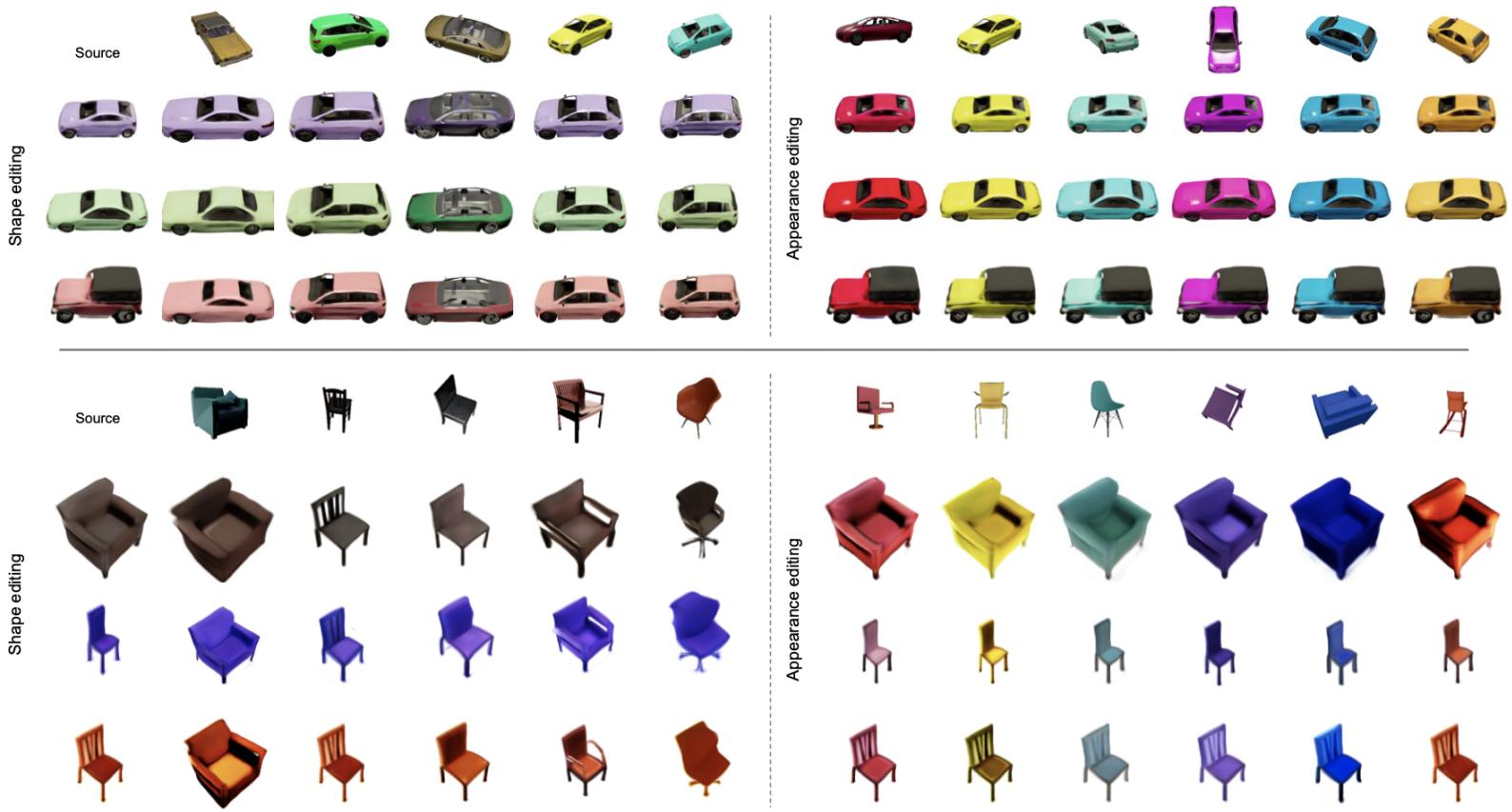


Figure 7. **Exemplar-Driven Editing Results.**

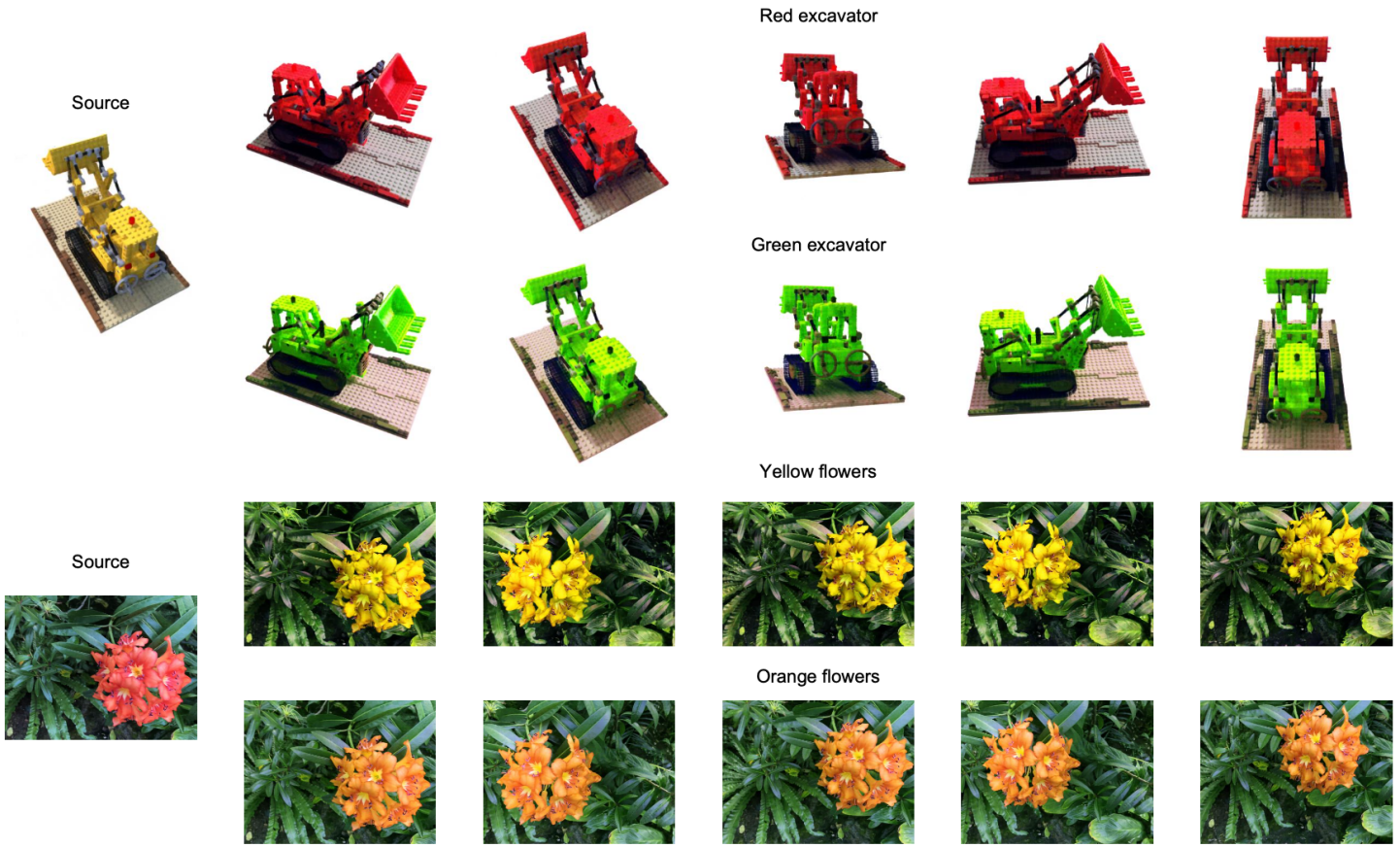


Figure 14. Single NeRF appearance editing results with our designed CLIP loss. NeRF models are trained on LLFF dataset [23].