CoNeRF: Controllable Neural Radiance Fields

Supplementary Material

A. Potential social impact

Our work is originally intended for creative and entertainment purposes, for example to allow users to easily edit their personal photos to have all the members of a group photo to have their eyes open. However, as with all work that enable editable models, our method has the potential to be misused for malicious purposes such as deep fakes. We strongly advise against such misuse. Recent work [6] has shown that it is possible to detect deep fakes, hinting that it should be possible to detect these deep learning-generated images. One of our future research direction is also along these lines, where we now aim to reliably detect images generated by our method.

B. Architecture details

We present architecture of: canonicalizer $\mathcal{K}$ in Fig. 10, attribute map $\mathcal{A}$ in Fig. 11, hypermap $\mathcal{H}$ in Fig. 12, per-attribute hypermap in Fig. 13, mask prediction network in Fig. 14 and the rendering network in Fig. 15. Each network contains only fully connected layers. Hidden layers use ReLU activation function. Colors of figures correspond to colors of blocks in Fig. 2b.

C. Additional qualitative results

See attached video clip for more qualitative results.

D. Failure Cases

We identify two modes of failure cases in our approach and present them in Fig. 9. In some cases with particular mask annotations, our model can struggle with controlling elements that occupy small space in the image. The problem is especially visible for controlling pendulum movement or opening and closing eyes. In the former, pendulum disappears and reappears in different places. In the latter, the control of eyes is periodic and there are two distant values in $[-1, 1]$ that produce opening eyes. While with careful annotations we noticed that the problem is mostly preventable, this problem may occur in practice.

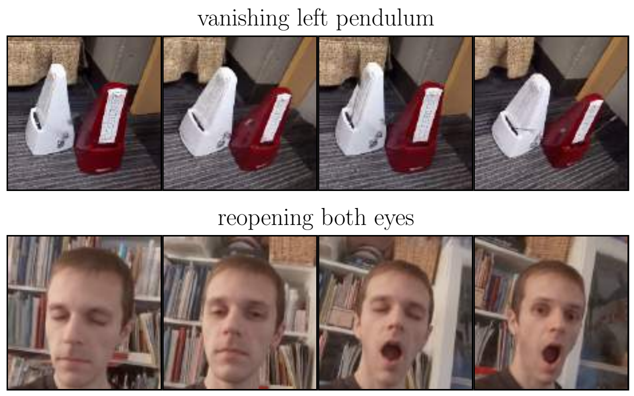


Figure 9. **Failure cases –** Our model may learn spurious interpolations for controlled elements that occupy little space in the image and with insufficient/careless annotations. For the metronome, due to the fast motion of the pendulum and its specularity, without careful annotation our method may simply learn its motion blur or sometimes even completely ignore the pendulum. In the face example, this may result in the eye blinking multiple times while interpolating between the attribute values of -1 and 1. Both cases are preventable with more careful annotations and by annotating more frames.

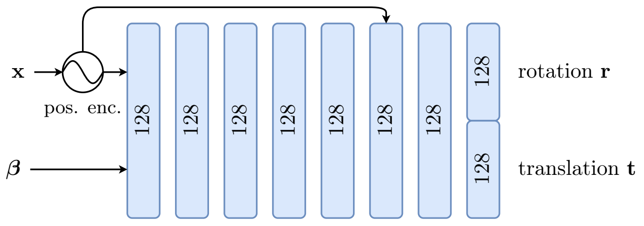


Figure 10. The canonicalization network takes positionally encoded raw coordinates $\mathbf{x}$ and learnable per-image latent code $\boldsymbol{\beta}$ and outputs rotation $\mathbf{r}$ expressed as a quaternion and translation $\mathbf{t}$. We rigidly transform each point $\mathbf{x}$ with an affine transform using both output. We use windowed positional encoding [36] for $\mathbf{x}$ with 8 components, linearly increasing contribution of components throughout 80k steps. We initialize the last layer to small values so the network can learn a base structure of the data.

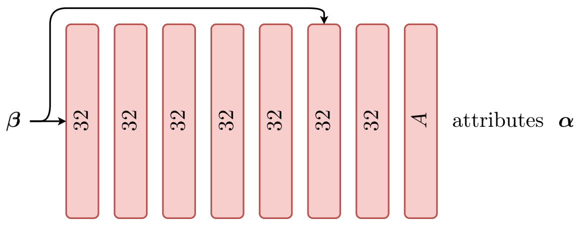


Figure 11. The attribute map $\mathcal{A}$ takes a per-image learnable latent code $\boldsymbol{\beta}$ and outputs A attributes $\boldsymbol{\alpha}$.

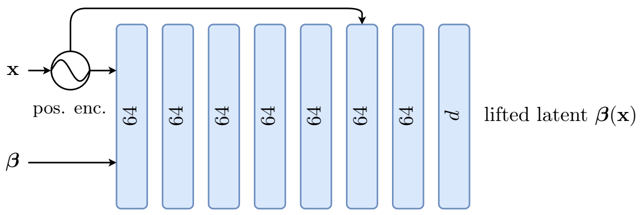


Figure 12. The network predicting lifted latent code $\boldsymbol{\beta}$, takes per-image $\boldsymbol{\beta}$ as an input, positionally encoded raw points $\boldsymbol{\beta}$ and outputs a lifted code of size d. We use only one sine component to encode $\mathbf{x}$.

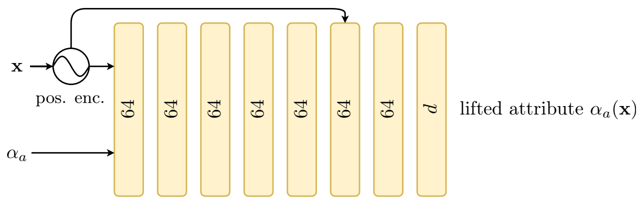
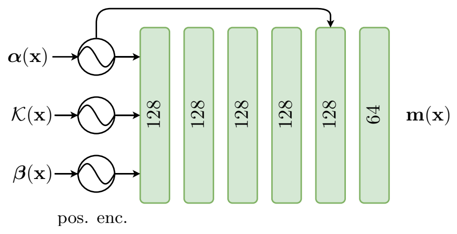
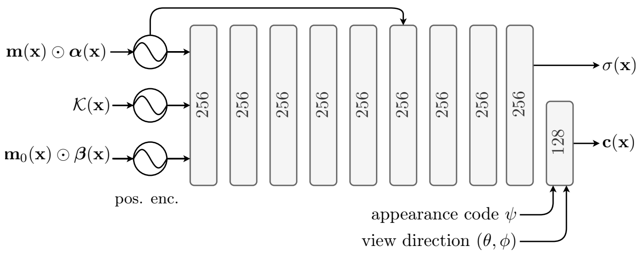


Figure 13. Per-attributes hypermaps take an attribute together with encoded $\mathbf{x}$ coordinates and output lifted $\alpha_a(\mathbf{x})$ ambient code of size d. We encode $\mathbf{x}$ with only single component.



$\boldsymbol{\alpha}(\mathbf{x})$
$\mathcal{K}(\mathbf{x})$
$\boldsymbol{\beta}(\mathbf{x})$
pos. enc.
128
64
$\mathbf{m}(\mathbf{x})$

Figure 14. Masking network $\mathcal{M}$ take lifted attributes $\boldsymbol{\alpha}(\mathbf{x})$, lifted latent code $\boldsymbol{\beta}(\mathbf{x})$ and canonicalized points $\mathcal{K}(\mathbf{x})$. We transform $\boldsymbol{\alpha}(\mathbf{x})$ and $\boldsymbol{\beta}(\mathbf{x})$ through a windowed positional encoding where we start at 1k-th step linearly increasing a single sine component for the next 10k steps. Points $\mathcal{K}(\mathbf{x})$ are encoded with 8 components. The output is activated with a sigmoid function. We realize $\mathbf{m}_0(\mathbf{x})$ as $\mathbf{m}(\mathbf{x})_0 = 1 - \sum_{a\in A} \mathbf{m}_a(\mathbf{x})$, and clip the output to ensure the values range to be in $[0, 1]$. Note that while the network shares similarities with the radiance field prediction part $\mathcal{R}$, it is not conditioned on view directions and appearance codes.



$\mathbf{m}(\mathbf{x}) \odot \boldsymbol{\alpha}(\mathbf{x})$
$\mathcal{K}(\mathbf{x})$
$\mathbf{m}_0(\mathbf{x}) \odot \boldsymbol{\beta}(\mathbf{x})$
pos. enc.
256
128
$\sigma(\mathbf{x})$
$\mathbf{c}(\mathbf{x})$
appearance code ψ
view direction (θ, ϕ)

Figure 15. The radiance field prediction network predicts RGB colors $\mathbf{c}(\mathbf{x})$ and density values $\sigma(\mathbf{x})$ from canonicalized points. We encode points $\mathbf{x}$ with 8 sine components and linearly increase contribution of a single component in $\boldsymbol{\alpha}(\mathbf{x})$ and $\boldsymbol{\beta}(\mathbf{x})$ from 1k to 11k step. Per-point predicted predicted attributes $\boldsymbol{\alpha}(\mathbf{x})$ and lifted latent code $\boldsymbol{\beta}(\mathbf{x})$ are masked by a mask predicted from the masking network depicted in Fig. 14. The final linear layer takes additional per-image learnable appearance code ψ to account for any visual variations that cannot be explained by the rest of the framework (*e.g.* changes in lighting). The code can discarded during evaluation. The same layer is additionally conditioned on the positionally encoded view directions. We activate the color output with a sigmoid function.