Supplementary Document for Sem2NeRF

Yuedong Chen[1]✉, Qianyi Wu[1], Chuanxia Zheng[1],
Tat-Jen Cham[2], and Jianfei Cai[1]

[1] Monash University, Australia
[2] Nanyang Technological University, Singapore
yuedong.chen@monash.edu

This supplementary document will provide additional technical details for *Sem2NeRF: Converting Single-View Semantic Masks to Neural Radiance Fields* in Section 1. Besides, more visual results for Sem2NeRF will be presented in Section 2. Unless otherwise specified, we refer to materials in the main paper [2] with a prefix "M-", *e.g.*, Fig. M-1 refers to Fig. 1 in the main paper.

1 Additional Technical Details

1.1 Builders for Additional Inputs

As mentioned in Section M-3.2, additional inputs, *i.e.*, contour and distance field representation, are provided for the encoder of Sem2NeRF to further highlight the boundary information. In this subsection, we will detail how to build these data using the python package "cv2"[3]. Note that both of them are directly calculated from the semantic mask, without involving any extra labels. And each builder function can be done in $1 \sim 2$ milliseconds.

Contour is generally represented as a curve, joining all the continuous points that share the same intensity. It is widely used as a tool to help shape analysis, object detection, *etc.*. In our implementation, we build the contour from the given one-hot encoded semantic mask. Main python codes are given as below. An output example is shown in Fig. 1 (middle).

```python
def binary_masks_to_contour(binary_masks):
    ''' INPUT: binary_masks, semantic mask in one-hot encoding form
        OUTPUT: contour, a contour map of the given semantic mask '''
    # initialize a black canvas
    mask = numpy.zeros((512, 512, 3), dtype=numpy.uint8)
    # find contours for each label
    for binary_mask in binary_masks:
        cnts = cv2.findContours(binary_mask, cv2.RETR_EXTERNAL,
                                cv2.CHAIN_APPROX_SIMPLE)
        cnts = cnts[0] if len(cnts) == 2 else cnts[1]
        for c in cnts:
            # draw contour with white color on the canvas
            cv2.drawContours(mask, [c], -1, (255, 255, 255), thickness=3)
    contour = cv2.cvtColor(mask, cv2.COLOR_BGR2GRAY)
    return contour
```

[3] https://pypi.org/project/opencv-python/

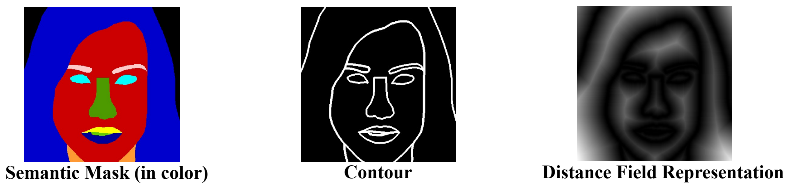

Fig. 1. An example of the encoder input. Semantic mask is shown in color for better visualization, while the network takes one-hot encoded mask as input

Distance field representation is a dense representation extracted from binary image via distance transformation. In the distance field, the grey intensity of each pixel indicates its distance to the nearest boundary. An unsigned Euclidean distance field representation is adopted in our experiments. Main python codes are given as below. An output example is shown in Fig. 1 (right).

```python
def contour_to_dist_filed(contour):
    ''' INPUT: contour, contour of the semantic mask
        OUTPUT: dist_field, distance filed representation '''
    # invert background and foreground of contour to match the setting
    invert_contour = cv2.bitwise_not(contour)
    dist_field = cv2.distanceTransform(invert_contour, cv2.DIST_L2, 3)
    # normalize the distance filed to [0, 1]
    cv2.normalize(dist_field, dist_field, 0, 1.0, cv2.NORM_MINMAX)
    dist_field = dist_field * 255.
    return dist_field
```

1.2 CatMask Dataset Rendering

To better demonstrate the introduced Semantic-to-NeRF translation task and evaluate the proposed Sem2NeRF framework, we develop a general solution to create datasets with pseudo labels using minimal human effort. In this work, we present an example by rendering a cat faces dataset, termed CatMask, which contains single-view cat faces generated by the pre-trained π-GAN model, pseudo ground-truth viewing directions, and 6-classes semantic masks labelled by DatasetGAN [6]. Similar to CelebAMask-HQ, CatMask only varies on the yaw and pitch axis. And it contains 28,000 training images and 2,000 testing images.

Cat faces from π-GAN. As mentioned in Section M-4.1, we assume the training data to have the viewing directions / poses of the single-view semantic-image pairs. However, different from human face, it is difficult to get the poses for cat faces. Considering that a pretrained π-GAN can generate photorealistic cat faces with given random poses, we choose to generate pseudo data for training our Sem2NeRF. Specifically, we randomly sample 30,000 vectors $\mathbf{z} \in \mathbb{R}^{256}$ from the input distribution of π-GAN to generate 30,000 corresponding cat faces

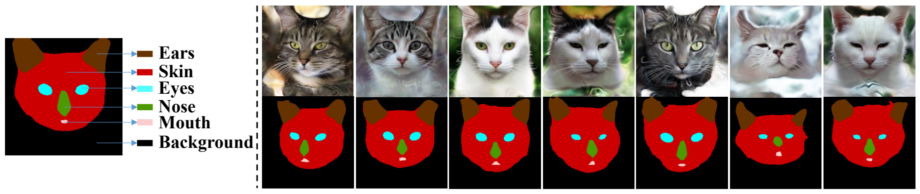

Fig. 2. CatMask dataset. Left: label legends of 6 semantic classes. Right: single-view cat faces rendered by π-GAN (top row) and the corresponding semantic masks labelled by DatasetGAN (bottom row). Best view in high quality color image

by using the released pretrained model[4]. Each image comes with one viewing direction, randomly sampled from normal distributions, with $X \sim \mathcal{N}(\pi/2,\, 0.3^2)$ for the yaw axis, $X \sim \mathcal{N}(\pi/2,\, 0.1^2)$ for the pitch axis, and 0 for the roll axis. Camera FOV is set to 18 to ensure the generated image covering the full cat face. Ray sampling resolution is set to 512×512, with ray depth range $[0.8,\, 1.2]$ and ray step size 72. Hierarchical sampling is enabled to improve image quality. We save the rendering viewing direction and the generated images for the CatMask dataset.

Cat semantic by DatasetGAN. A suitable training dataset for Sem2NeRF should be able to be modelled by existing NeRF-based generator, while also comes with semantic labels. However, most datasets used by existing 3D-aware generative models, *e.g.*, cat and car, do not contain component-level semantic masks. We further find out that DatasetGAN can create reasonable semantic masks labels for our task.

DatasetGAN is introduced to automatically build datasets of high-quality semantically segmented images. Specifically, it proposes a MLP-based "Style Interpreter" that can be trained to decode the feature maps of a pretrained StyleGAN model to semantic labels, requiring only very few manually-labeled training samples, *e.g.*, 30 labelled images for cat dataset. In this case, dataset can be automatically built by first randomly sampling images from StyleGAN, following by parsing with the trained style interpreter to obtain corresponding semantic labels.

We use the released[5] pretrained style interpreter for cat to generate a dataset of 10,000 images-semantic pairs. Such a dataset is further leveraged to train a Deeplab-V3 [1] model, which takes a cat image as input and outputs the corresponding cat semantic mask. We then use the trained Deeplab-V3 to label our generated CatMask dataset. 6 classes are selected based on the label quality. Label legends and examples of the CatMask dataset are given in Fig. 2.

[4] `https://github.com/marcoamonteiro/pi-GAN`
[5] `https://github.com/nv-tlabs/datasetGAN_release`

1.3 Additional Implementation Details

Style codes averages ($\overline{\gamma}, \overline{\beta}$). We randomly sample 10,000 vectors $\mathbf{z} \in \mathbb{R}^{256}$ from a standard normal distribution, then feed them through the pretrained mapping network of the origianl π-GAN model, finally average the outputs over the batch dimension to obtain $\overline{\gamma}, \overline{\beta}$.

Datasets. For CelebAMask-HQ [4], both images and semantic masks are loaded as resolution 640×640, then center cropped to 512×512. For CatMask, images and semantic masks are directly loaded as 512×512. Semantic masks are transformed using one-hot encoding, augmented with the aforementioned contours and distance field representations. We do not apply any other data augmentation, *e.g.*, random flip, to avoid harming the pose information.

Training. Patch discriminator with input size 128×128 is adopted in our experiments, using the implementation provided by the GRAF [5] project[6]. Images are rendered via only the "coarse" network of the decoder, *i.e.*, removing the hierarchical sampling. The sampling range of the scaling factor α of Eq. M-(2) is initialized as [0.9, 1.0], where the lower bound is exponentially annealed to 0.06 during training. Encoder is initialized with the ImageNet-1K [3] pretrained weights, decoder is initialized with π-GAN pretrained weights, while the discriminator is randomly initialized. We freeze the parameters of the decoder, and set the learning rate of the encoder and discriminator to 1×10^{-4} and 2×10^{-5}, respectively. Ranger optimizer[7] is used for both encoder and discriminator. We set the training batch size to 8, and use V100 GPUs to train all related models for 200,000 iterations.

Inference. To render qualitative results, rays are cast with size 512×512 and depth step 72 in the inference. Besides, "fine" network is activated to enable hierarchical sampling. For GAN-inversion used by pix2pixHD as mentioned in Section M-4.1, we adopt the implementation from the π-GAN project[8], and set the iteration number to 700 as suggested.

2 Additional Visual Results

In the following three pages, we will present additional visual results for the proposed Sem2NeRF regarding free-viewpoint image generation (see Section 2.1), semantic mask editing (see Section 2.2) and multi-modal synthesis (see Section 2.3). Results are demonstrated on both CelebAMask-HQ and CatMask, and they are all best viewed in high quality color image.

[6] `https://github.com/autonomousvision/graf`
[7] `https://github.com/lessw2020/Ranger-Deep-Learning-Optimizer`
[8] `https://github.com/marcoamonteiro/pi-GAN`

2.1 Free-viewpoint Image Generation

Additional visual results of free-viewpoint image generation on CelebAMask-HQ and CatMask datasets are given in Fig. 3 and Fig. 4, respectively.

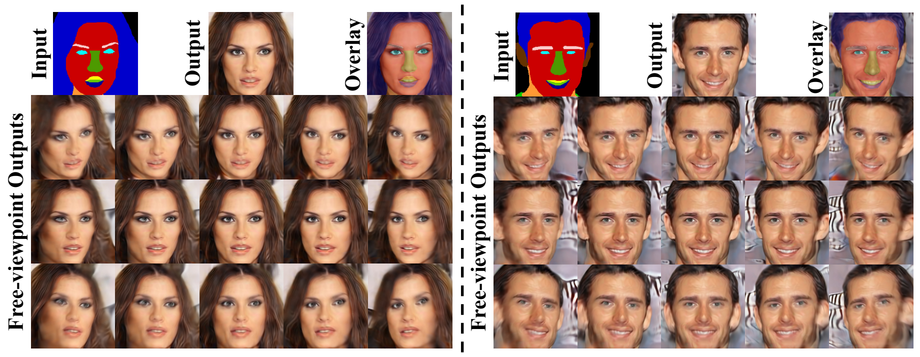

Fig. 3. Free-viewpoint image generation on CelebAMask-HQ. "Output" refers to the generated image that has the same viewing direction as the "Input", and "Overlay" shows the results of overlaying "Output" with "Input", so as to better demonstrate the mapping accuracy

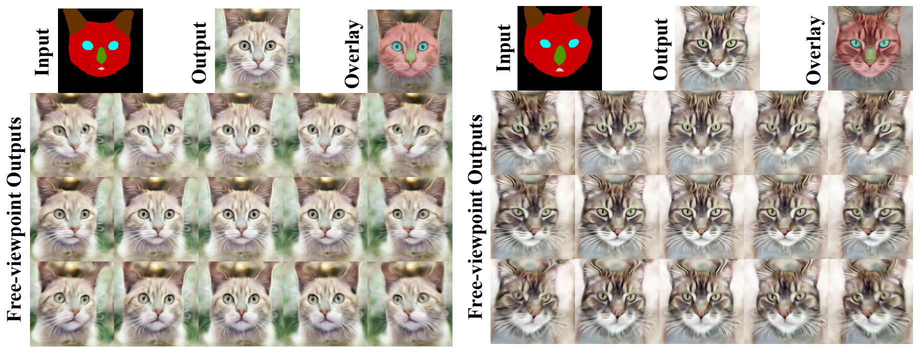

Fig. 4. Free-viewpoint image generation on CatMask. "Output" refers to the generated image that has the same viewing direction as the "Input", and "Overlay" shows the results of overlaying "Output" with "Input", so as to better demonstrate the mapping accuracy

2.2 Semantic Mask Editing

Additional visual results of semantic mask editing on CelebAMask-HQ and CatMask datasets are given in Fig. 5 and Fig. 6, respectively.

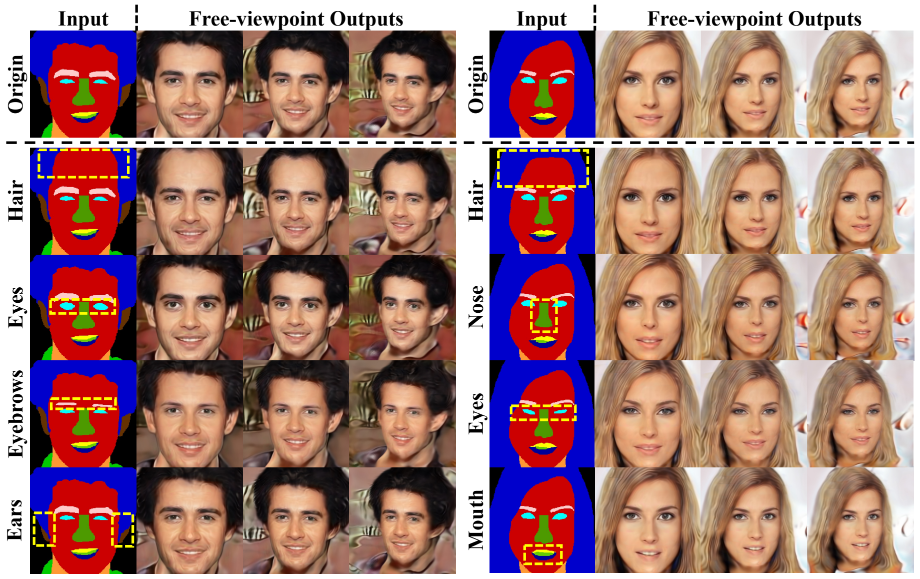

Fig. 5. Semantic mask editing on CelebAMask-HQ. The first row shows the results of the original semantic masks, while the following rows give the results of editing the mentioned area, highlighted with yellow-dash box. Three viewpoints are given for each group, with the first one having the same viewing direction as the input

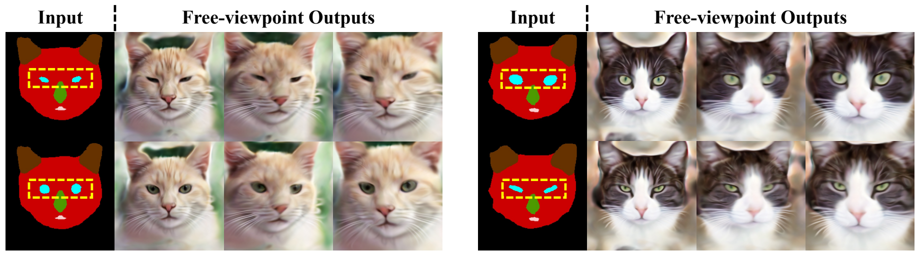

Fig. 6. Semantic mask editing on CatMask. Edited regions are highlighted with yellow-dash box. The first viewpoint has the same pose as the input

2.3 Multi-modal Synthesis

Additional visual results regarding multi-modal synthesis on CelebAMask-HQ and CatMask datasets are given in Fig. 7 and Fig. 8, respectively.

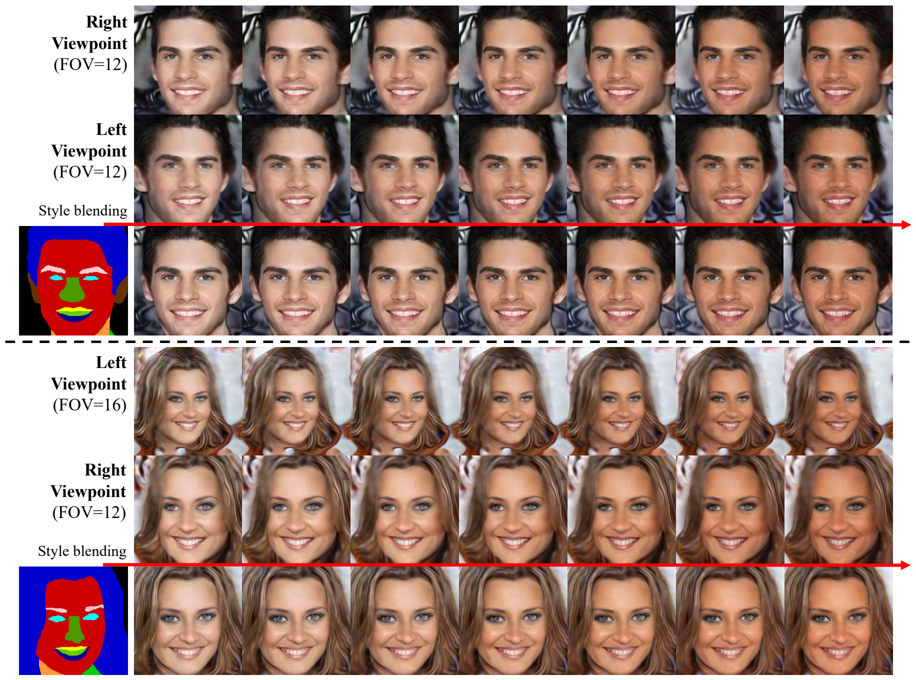

Fig. 7. Multi-modal synthesis on CelebAMask-HQ. Styles are linearly blended from left to right. The last viewpoint in each group has the same pose as the input

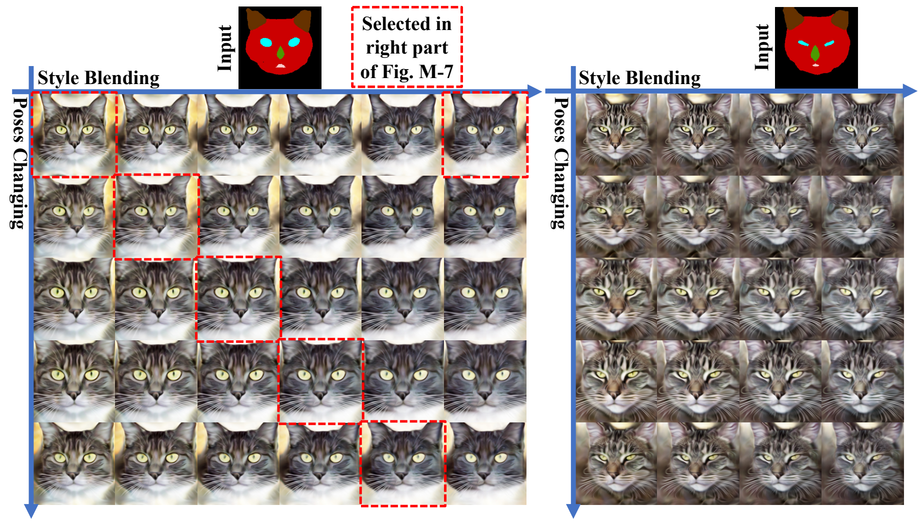

Fig. 8. Multi-modal synthesis on CatMask. Left case shows the full version of Fig. M-7 (right part), where the selected images are highlighted in red-dash border. Images in the first row have the same viewing direction as the input

References

1. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans. Pattern Anal. Mach. Intell. **40**(4), 834–848 (2017)
2. Chen, Y., Wu, Q., Zheng, C., Cham, T.J., Cai, J.: Sem2nerf: Converting single-view semantic masks to neural radiance fields. arXiv preprint arXiv:2203.10821 (2022)
3. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 248–255. Ieee (2009)
4. Lee, C.H., Liu, Z., Wu, L., Luo, P.: Maskgan: Towards diverse and interactive facial image manipulation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5549–5558 (2020)
5. Schwarz, K., Liao, Y., Niemeyer, M., Geiger, A.: Graf: Generative radiance fields for 3d-aware image synthesis. Adv. Neural Inform. Process. Syst. **33**, 20154–20166 (2020)
6. Zhang, Y., Ling, H., Gao, J., Yin, K., Lafleche, J.F., Barriuso, A., Torralba, A., Fidler, S.: Datasetgan: Efficient labeled data factory with minimal human effort. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 10145–10155 (2021)