Supplementary Material of Foreground-aware Image Inpainting

1. Network Architecture

In this section, we introduce the detailed configuration of our networks. Our model is composed of three modules, the contour detection module, the contour completion module and the image completion module.

1.1. Contour Detection Module

The key component of our contour detection module is the saliency object segmentation network. We describe the details of this network here. The segmentation network used in our paper consists of three major parts: High-level Stream, Low-level Stream and Boundary Refine Module.

High-level Stream It takes the incomplete image as input and uses the encoder part of a traditional segmentation network to extract compact features. The output is a two-channel low resolution feature map, which is used as the bottle-neck of the network. In this module, we use Inception V2 as the segmentation network. The input of the network is a 3-channel image and the original output of the truncated Inceptions-V2 is a 7x7 1024-channel feature map. In order to get a 14x14 feature map, we use dilated convolution for the last two inception modules. Finally, we add a convolution layer to generate the 2-channel 14x14 feature map.

Low-level Stream This module is a shallow network composed of a single 7x7 convolution layer with a stride of 1. The input to the shallow network is our incomplete image. The output of this stream is a 64-channel feature map that has the same spatial size as the input image.

Boundary Refine Module This module takes the low-level and high-level feature as input and outputs the final result. Specifically, we first resize the high-level feature map to the original resolution by bilinear upsampling. Then, we concatenate the upsampled high-level feature map with the low-level feature map and pass them to the densely connected layer units. Each dense unit is composed of some convolutional layers, and the output will be concatenated with the input to the unit.

1.2. Contour Completion Module

Our contour completion module shares a similar architecture with GatedConv [1]. Specifically, it consists of two stages. The first stage is a encoder-decoder network that takes the incomplete contour, the incomplete image and the mask as inputs, and outputs a coarse result of the completed contour. The encoder is a cascade of several gated convolution blocks described in [1], and finally maps the input image to feature maps with a spatial resolution of 64x64. The decoder has a reverse architecture as the encoder and maps the feature maps to a completed contour image. The coarse contour is then concatenated with the mask and then input to the refine network of the contour completion module, to get the final result. The refine network has a two-stream encoder that maps the inputs to feature maps of size 64x64, and a decoder that maps the feature maps to the final image.

The detailed configuration of the contour completion module is as follows. For simplicity, we denote kernel size, dilation, stride size and channel number as K, D, S, C, respectively.

Coarse Network: K5S1C48 - K3S2C96 - K3S1C96 - K3S2C192 - K3S1C192 - K3S1C192 - K3D2S1C192 - K3D4S1C192 - K3D8S1C192 - K3D16S1C192 - K3S1C192 - K3S1C192 - resize (2) - K3S1C96 - K3S1C96 - resize (2) - K3S1C48 - K3S1C24 - K3S1C3 - sigmoid.

Refine Network:

Branch-1: K5S1C48 - K3S2C96 - K3S1C96 - K3S2C192 - K3S1C192 - K3S1C192 - K3D2S1C192 - K3D4S1C192

Branch-2: K5S1C48 - K3S2C948 - K3S1C96 - K3S2C192 - K3S1C192 - K3S1C192 (contextual attention) - K3S1C192 - K3S1C192

Decoder: concat - K3S1C192 - K3S1C192 - resize (2) - K3S1C96 - K3S1C96 - resize (2) - K3S1C48 - K3S1C24 - K3S1C3 - sigmoid.

1.3. Image Completion Module

The image completion module has the same architecture as the contour completion module, except for the inputs and outputs to each network. The input to the refine network is the coarse completed image, the completed contour and the mask, the output of the coarse network and the refine network is activated with tanh function, instead of sigmoid which is used in the contour completion module.

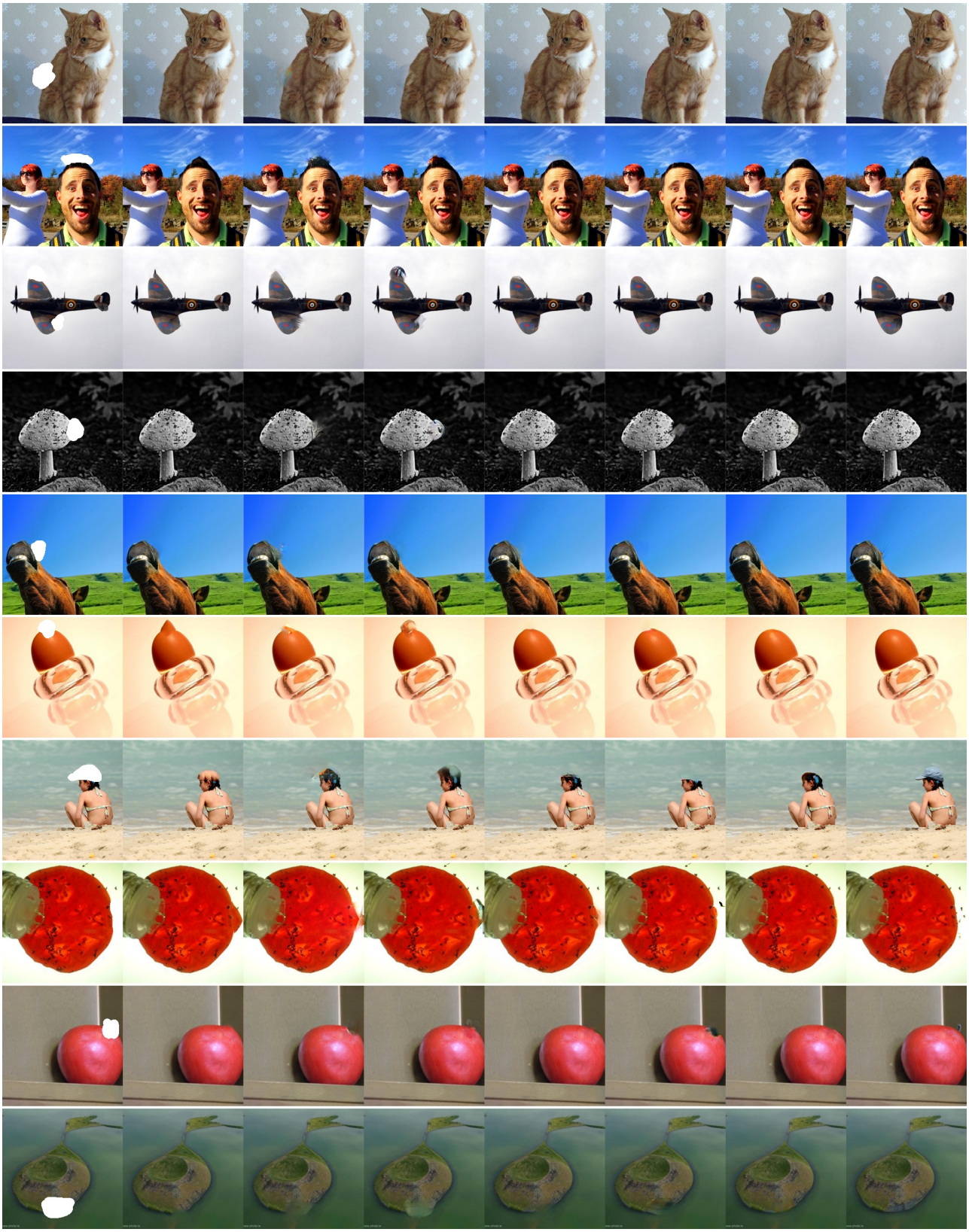

Figure 1. Qualitative comparison between the state-of-the-art methods. Please zoom in to see the details.

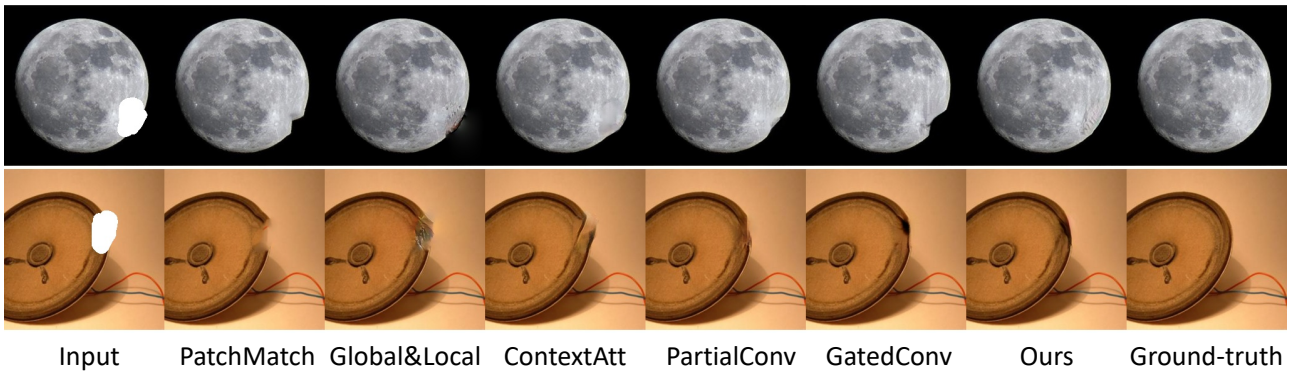

Figure 2. Failure cases. Please zoom in to see the details.

2. Comparison with State-of-the-arts

2.1. Qualitative Results

In this section, we show more qualitative results. As can be seen from Fig. 1, our model consistently outperforms the state-of-the-art models.

2.2. Quantitative Results

To make a more thorough comparison, we also include the results for each model using Perceptual Similarity LPIPS [2] on the feature space of VGG or AlexNet, and the results are shown in Table 1.

Table 1. Additional quantitative metrics, smaller is better.

Method	LPIPS (VGG)	LPIPS (Alex)
PConv	0.064	0.044
GatedConv	0.063	0.048
Ours Guided	**0.060**	**0.043**

3. Contour Completion Results

We supplement more contour completion results here. As is shown in Fig. 3, our contour completion module can infer clean, sharp and reasonable contours, which can be of great benefits to the completion of the image.

4. Failure Cases

In this section, we show some cases that the existing models fail to inpaint. The results are shown in Fig. 2. Seen from the figure, though our model is able to complete a reasonable contour for the incomplete object, however, sometimes, artifacts can still occur. In our future work, we will try to reduce the artifacts while predicting a reasonable shape for the objects.

Figure 3. Contour completion results. From left to right: image with hole, saliency map of the incomplete image, incomplete contour, completed contour and the completed image.

References

[1] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. *arXiv preprint arXiv:1806.03589*, 2018.

[2] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.