Supplementary Material: Conditional Image Repainting via Semantic Bridge and Piecewise Value Function

A Appendix

A.1 The optimality of $D(\bar{y}|y)$

To supplement §3.2 at L206-208, we provide the following proof.
Proposition. For G fixed, the optimal discriminator $D(\bar{y}|y)$ is $D^*_G(\bar{y}|y) = 0$.
Proof. The training criterion for the discriminator $D(\bar{y}|y)$, given any generator G, is to maximize the quantity $V(D,G)$[1]

$$\begin{aligned} V(D,G) = &\int_y p_{\text{data}}(y)\log(1-D(\bar{y}|y)) + p_{\text{g}}(y)\log(1-D(\bar{y}|y))dy \\ &\int_y (p_{\text{data}}(y) + p_{\text{g}}(y))\log(1-D(\bar{y}|y))dy \end{aligned} \tag{1}$$

For any $(a,b) \in \mathbb{R}^2 \backslash \{0,0\}$, the maximum of the function $y \to (a+b)\log(1-y)$ has nothing to do with the values of (a,b), but it is achieved when $y = 0$. The discriminator does not need to be defined outside of $\text{supp}(p_{\text{data}}) \cup \text{supp}(p_{\text{g}})$, concluding the proof.

A.2 Supplemental network architecture design

Conditional content generator G^{cg}. The overall architecture is shown in Fig. 1 in which we visualize two stages. In practice, we implement G^{cg} to be three-stage, which synthesizes 256×256 images. The implementation of the geometry encoder Enc^{g} and texture encoder Enc^{t} is shown in Fig. 2. For the color encoder Enc^{c}, we directly adopt the bidirectional LSTM based text encoder used in [25]. We also use the same training mechanism as in [25] for training Enc^{c}. The GAIN ResBlk is implemented as in Fig. 3(a).
Gated Adaptive INstance (GAIN) normalization (Fig. 3(b)) is designed based on AdaIN [10] (Fig. 3(c)) and SPADE [13]. We also visualize the implementation of *SPADE w/ UniTex* (Table 1) in Fig. 3(d).
Compositing model G^{cc} is implemented as in Fig. 4(a). We also show the implementation of the image-to-image translation baseline *Pix2pix* in Fig. 4(b).
Discriminators are designed as in Fig. 5. Their functionalities in supervising the generation model G^{cg} and the compositing model G^{cc} are presented in §A.3. The comparison between the proposed classification based compositing discriminator D^{cc} (§4.3) and the segmentation based discriminator (§4.2) can be found in Fig. 5(c) and (d). Their convergence comparison is presented in §A.3.

[1] For the notion simplification, we omit the mean-reduction function ξ used in (3).

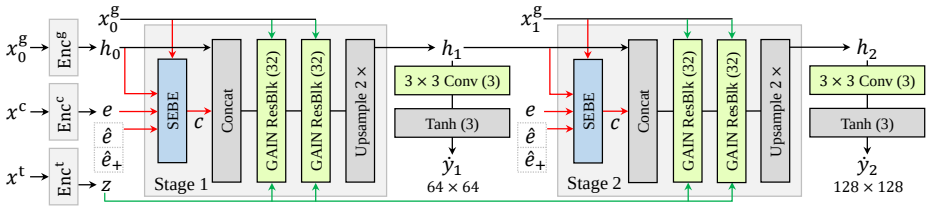

Fig. 1. The multistage conditional content generator G^{cg}. Each stage outputs an image $\dot{y}_i$ of a specific resolution. Parentheses in module names enclose the number of output channels of a module. "Upsample 2×" means upsampling the resolution of a feature map by 2 times.

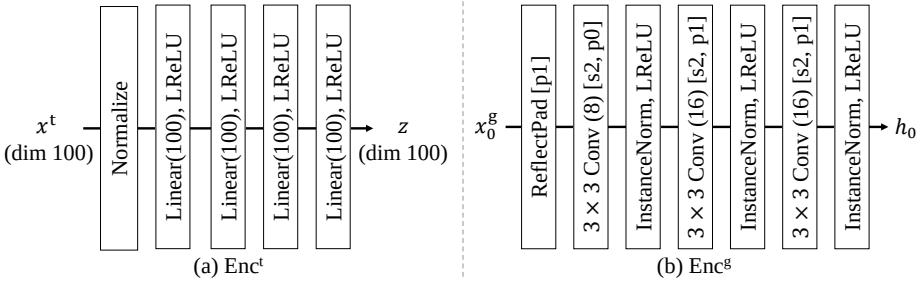

Fig. 2. *(a)* The encoder for the input texture condition. *(b)* The encoder for the input geometry condition. The square brackets enclose the stride and padding information of the convolutional layers.

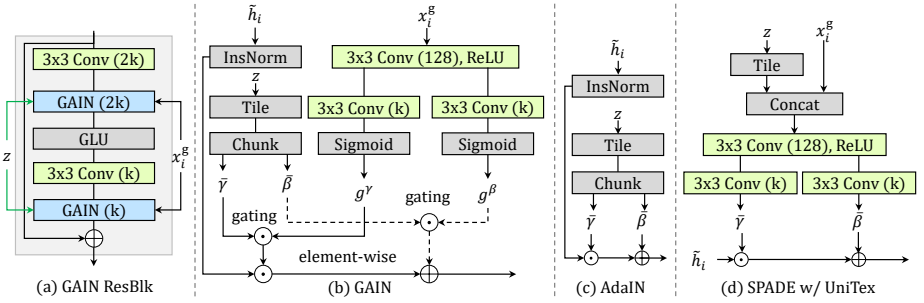

Fig. 3. *(a)* GAIN ResBlk in Fig. 1, where GLU is abbreviated for Gated Linear Unit. *(b)* GAIN based on AdaIN [10] (left) and SPADE [13] (right), where $\tilde{h}_i$ denotes the intermediate feature maps. *(c)* AdaIN [10] in Table 1. *(d)* SPADE w/ UniTex in Table 1.

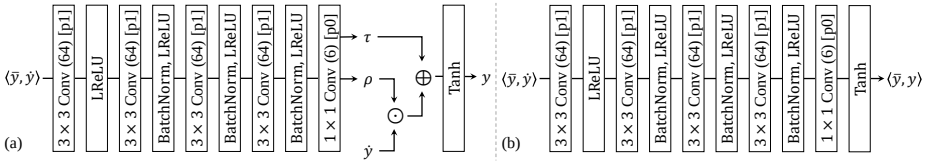

Fig. 4. *(a)* Compositing model G^{cc} for Category Seg and Cls in Table 1, where (ρ, τ) denote the inferred affine parameters. *(b)* Compositing model for Pix2pix in Table 1.

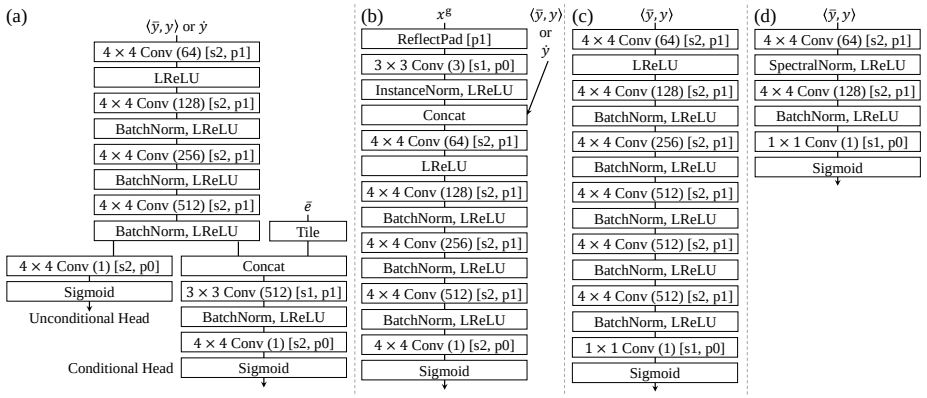

Fig. 5. *(a)* Joint-conditional-unconditional patch discriminators and *(b)* shape discriminators in [11], where $\langle \bar{y}, y \rangle$ represents the composition of the innate content $\bar{y}$ and the transformed generated content y. *(c)* Patch-wise classification based discriminators D^{cc} in §4.3, and also those for Category Cls in Table 1. *(d)* Segmentation-based adversarial discriminator mentioned in §3.2, and also those for Category Seg in Table 1.

A.3 Supplemental learning

Losses. We present the overall minimax optimization problem in (10), which includes four losses $\mathcal{L}_{\text{cg}}$, $\mathcal{L}_{\text{cm}}$, $\mathcal{L}_{\text{cc}}$ and $\mathcal{L}_{\text{r}}$. We define $\mathcal{L}_{\text{cc}}$ in (11), and directly adopt the DAMSM loss $\mathcal{L}_{\text{cm}}$ in [25]. So, we focus on introducing $\mathcal{L}_{\text{cg}}$ and $\mathcal{L}_{\text{r}}$.

In Eq. (10), $D^{\text{cg}} = \{(D_1^{\text{pat}}, D_1^{\text{shp}}), \ldots, (D_i^{\text{pat}}, D_i^{\text{shp}}), \ldots, (D_n^{\text{pat}}, D_n^{\text{shp}})\}$ is a set of discriminators [11] for each stage of G^{cg}, where D_i^{pat} is a joint-conditional-unconditional patch discriminator (conditioned on the color condition), and D_i^{shp} is a discriminator conditioned on the geometry condition. D_i^{pat} and D_i^{shp} are elaborated in detail in [11], so we directly borrow their symbols to ease readers' references.

Given a pair of image and color condition (global sentence embedding), *i.e.*, $(y, \bar{e})$, D_i^{pat} can be written as: $\mathbf{p}^{\mathsf{u}}[y]_i = D_i^{\text{pat}}(y)$, $\mathbf{p}^{\mathsf{c}}[y, \bar{e}]_i = D_i^{\text{pat}}(y, \bar{e})$, where the superscript u and c indicate the "unconditional" and "conditional". $\mathbf{p} = \{p_1, \ldots, p_j, \ldots, p_{N^{\text{pat}}}\}$ is a set of probabilities with each indicating the realness of a patch. The input to D_i^{pat} is indicated within the square brackets. Given

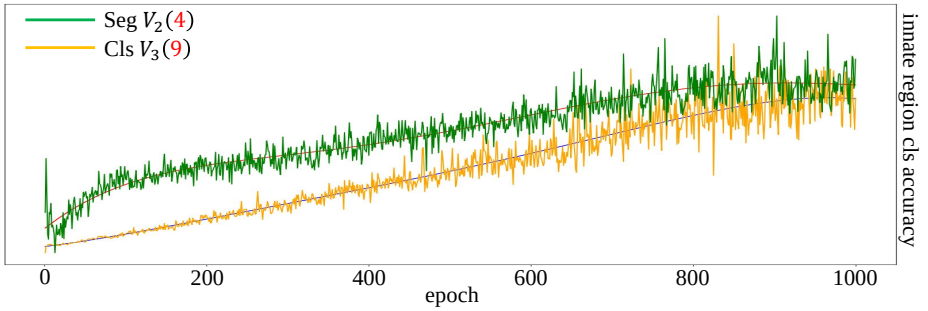

Fig. 6. Pixel-wise segmentation accuracy vs. patch-wise classification accuracy for regions of the innate content. The absolute values of these two accuracies are incomparable because of different settings, so they are adjusted to a similar scale for better revealing the comparison in the convergence speed between the segmentation-based discriminator Seg $V_2(4)$ and the proposed classification-based discriminator Cls $V_3(9)$.

a pair of image and geometry condition, *i.e.*, (y, x^{g}), D_i^{shp} can be written as: $\mathbf{p}^{\mathtt{g}}[y, x^{\mathrm{g}}]_i = D_i^{\mathrm{shp}}(y, x^{\mathrm{g}})$ where the superscript $\mathtt{g}$ indicate the conditioning on the geometry. $\mathcal{L}_{\mathrm{cg}}$ helps both G^{cg} and G^{cc} to produce realistic images, which is defined as follows:

$$\begin{aligned}
&\mathcal{L}_{\mathrm{cg}}(G^{\mathrm{cg}}, G^{\mathrm{cc}}, D^{\mathrm{cg}}) = \\
&-\sum_{i=1}^{n} \frac{1}{2N_i^{\mathrm{pat}}} \sum_{j=1}^{N_i^{\mathrm{pat}}} \left(\lambda^{\mathtt{u}} \log p_j^{\mathtt{u}}[\dot{y}_i]_i + \log p_j^{\mathtt{c}}[\dot{y}_i, \bar{e}]_i + \log p_j^{\mathtt{g}}[\dot{y}_i, x_i^{\mathrm{g}}]_i\right) \\
&\quad -\frac{1}{2N_n^{\mathrm{pat}}} \sum_{j=1}^{N_n^{\mathrm{pat}}} \left(\lambda^{\mathtt{u}} \log p_j^{\mathtt{u}}[\langle \bar{y}, y\rangle]_n + \log p_j^{\mathtt{c}}[\langle \bar{y}, y\rangle, \bar{e}]_n + \log p_j^{\mathtt{g}}[\langle \bar{y}, y\rangle, x_n^{\mathrm{g}}]_n\right),
\end{aligned} \tag{2}$$

where $\dot{y}_i$ is the generated image at the i-th stage of G^{cg}, and $\dot{y}_i$ is transformed by G^{cc} to form y_i. $\langle \bar{y}, y\rangle$ represents an image composited of the innate content $\bar{y}$ and the transformed generated content y. $\lambda^{\mathtt{u}}$ is a balancing hyperparameter which is set to 4.0.

$\mathcal{L}_{\mathrm{r}}$ is a pixel-wise L1 loss to regularize the training so as to anchor the transformed generated content y to the original $\dot{y}$ which is generated by G^{cg}. $\mathcal{L}_{\mathrm{r}}$ is defined as follows:

$$\mathcal{L}_{\mathrm{r}}(F^{\mathrm{C}}) = \frac{1}{N^{\mathrm{fpix}}} \sum_{j=1}^{N^{\mathrm{fpix}}} |y[j] - \dot{y}[j]|\,, \tag{3}$$

where N^{fpix} denotes the number of composited pixels. $y[j]$ and $\dot{y}[j]$ are the j-th pixel of an image.

Convergence comparison. In Fig. 6, we visualize the accuracy evolution throughout the training process for the segmentation-based discriminator Seg $V_2(4)$ and the proposed classification-based discriminator Cls $V_3(9)$. It shows

that Seg V_2(4) (green in Fig. 6) becomes successful much faster than Cls V_3(9) (yellow in Fig. 6). This demonstrates the effectiveness of our proposed compositing discriminator D^{cc} in impeding the convergence, which paves the way for using piecewise value function (8) to improve the compositing performance (see Limitation 2).

A.4 Supplemental qualitative study

Alternation of input conditions. To supplement Fig. 4, we provide more bird results by alternating input conditions in Fig. 7 and 8.
Comparison with the modified GauGAN. As mentioned in §5.3, by directly incorporating SEBE and the proposed compositing techniques into GauGAN [13], it cannot effectively vary the gray-scale textures. This conclusion can be reached by comparing Fig. 8 (our results) and Fig. 9 (results of the modified GauGAN). This is because in GauGAN, the gray-scale texture condition (a Gaussian noise) is provided at the very beginning of the network, which might be washed away by the normalization layers.
Iterative image editing in the wild. To supplement Fig. 6, we provide more results in Fig. 10, 11, 12, 13 and 14.
Object removal. To supplement Fig. 5, we provide Fig. 15 and 16. The results show that the inpainting method [27] tends to yield artifacts for cases where the image is a clutter. We observe that our method can achieve great removal performance if given an entire object mask covering the object to be removed (right column in Fig. 15 and 16). However, we also observe that if only given a precise mask enclosing the object to be removed, it is difficult for our method to make the generated content indistinguishable from the innate content of the same class, *e.g.*, sky in Fig. 15 and lawn in Fig. 16. This might be because the proposed generation and compositing techniques do not condition on the innate content of the same class. We leave this sub-direction for future exploration.

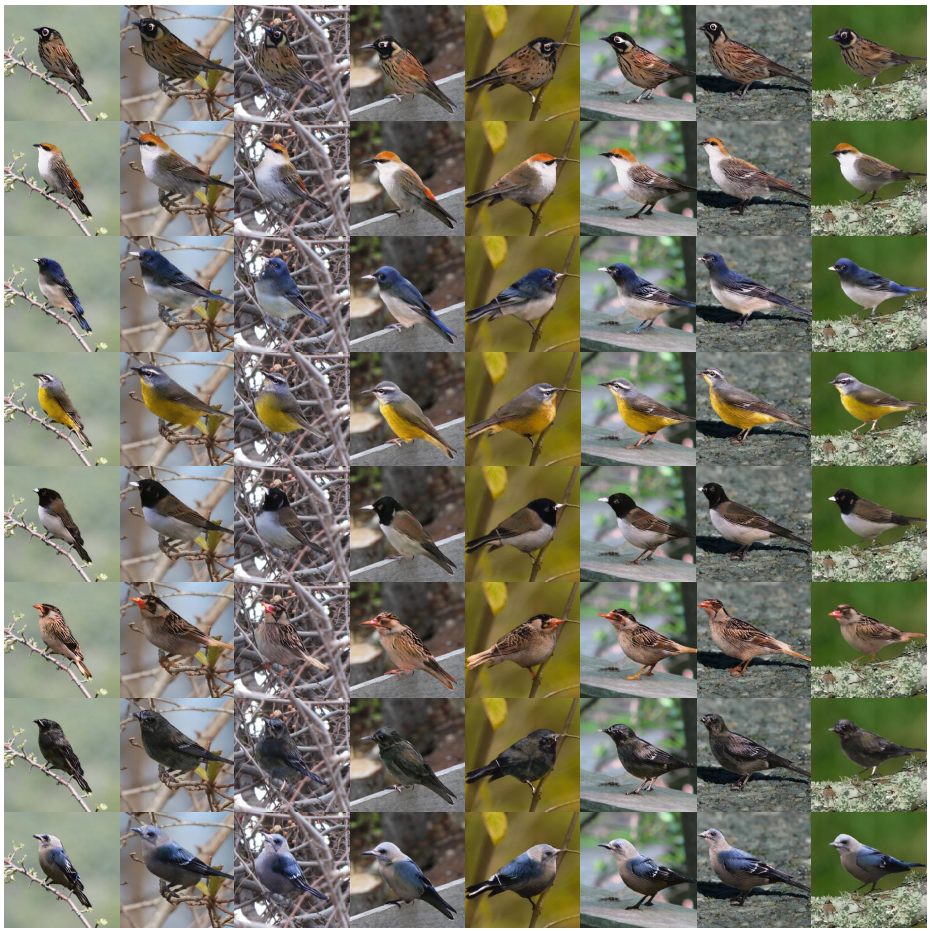

Fig. 7. Alternation of input geometry and color conditions for birds.

Fig. 8. Alternation of input gray-scale texture condition for birds.

Fig. 9. Alternation of input gray-scale texture condition for birds, given by directly incorporating SEBE and the proposed compositing techniques into GauGAN [13].

Fig. 10. Alternation of input conditions in the wild. Column 1 show the real images, and Columns 2-4 show the results by alternating input conditions.

Fig. 11. Alternation of input conditions in the wild. Column 1 show the real images, and Columns 2-4 show the results by alternating input conditions.

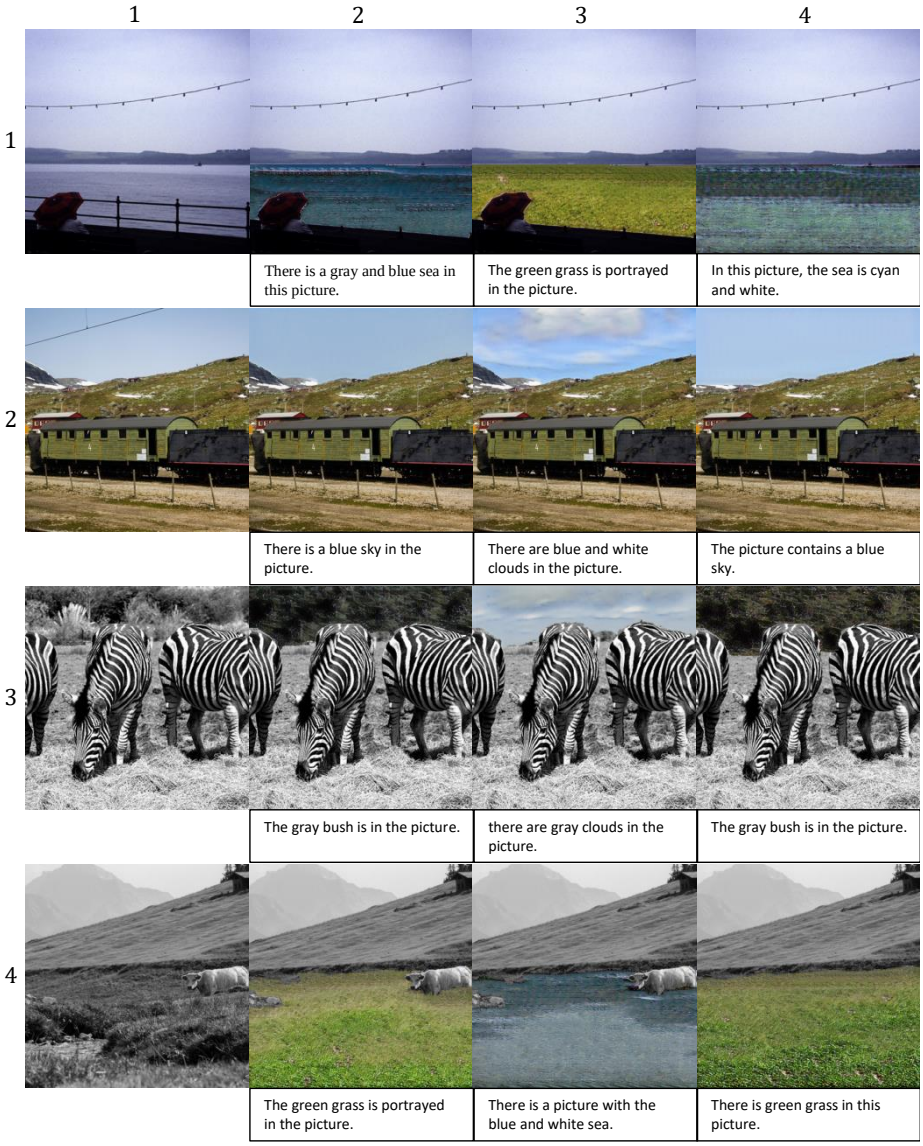

Fig. 12. Alternation of input conditions in the wild. Column 1 show the real images, and Columns 2-4 show the results by alternating input conditions.

Fig. 13. Iterative editing in the wild. Column 1 show the real images, and Columns 2-4 show the iterative editing.

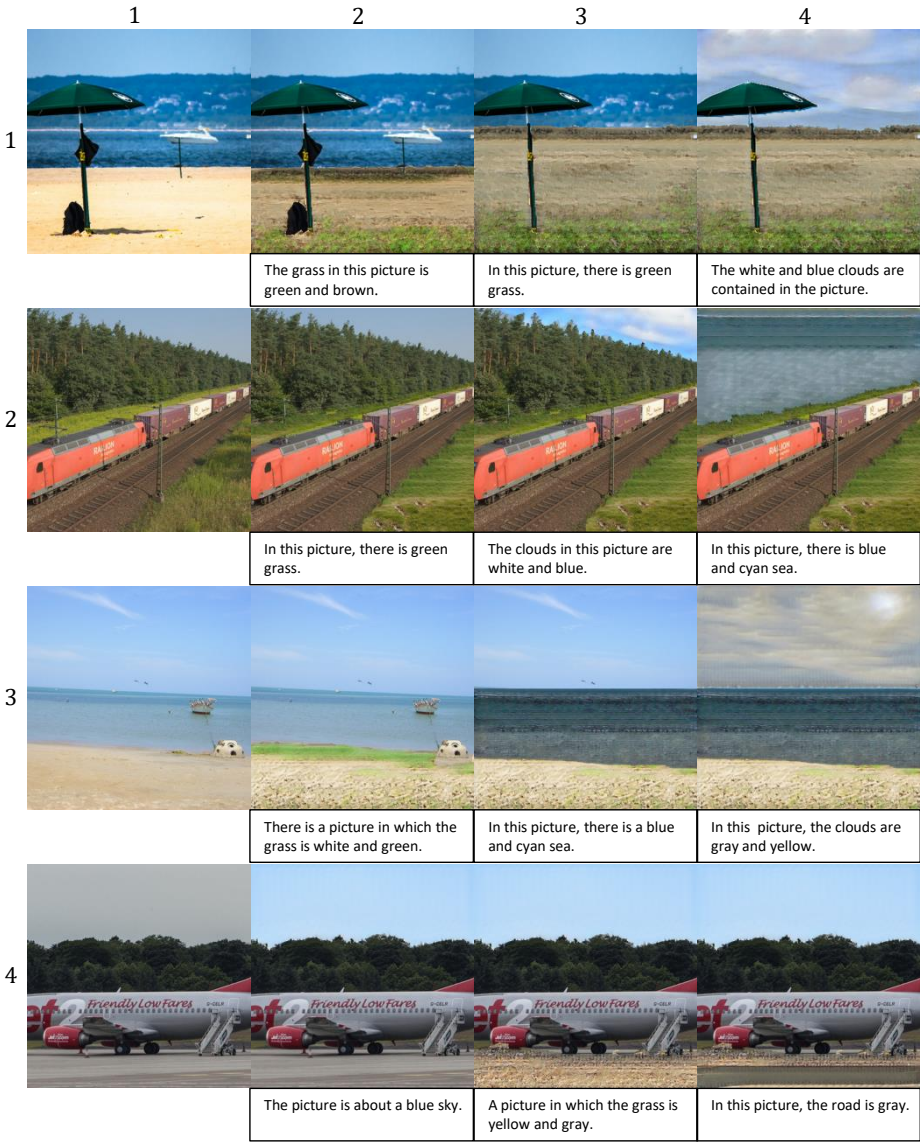

Fig. 14. Iterative editing in the wild. Column 1 show the real images, and Columns 2-4 show the iterative editing.

Fig. 15. Comparison with [27] for mask based object removal. Masks are shown at Row 1, where the gray indicates regions to be filled. Column 1 shows the real image. Row 2 and Row 3 show results edited following the given masks by [27] and our method, respectively.

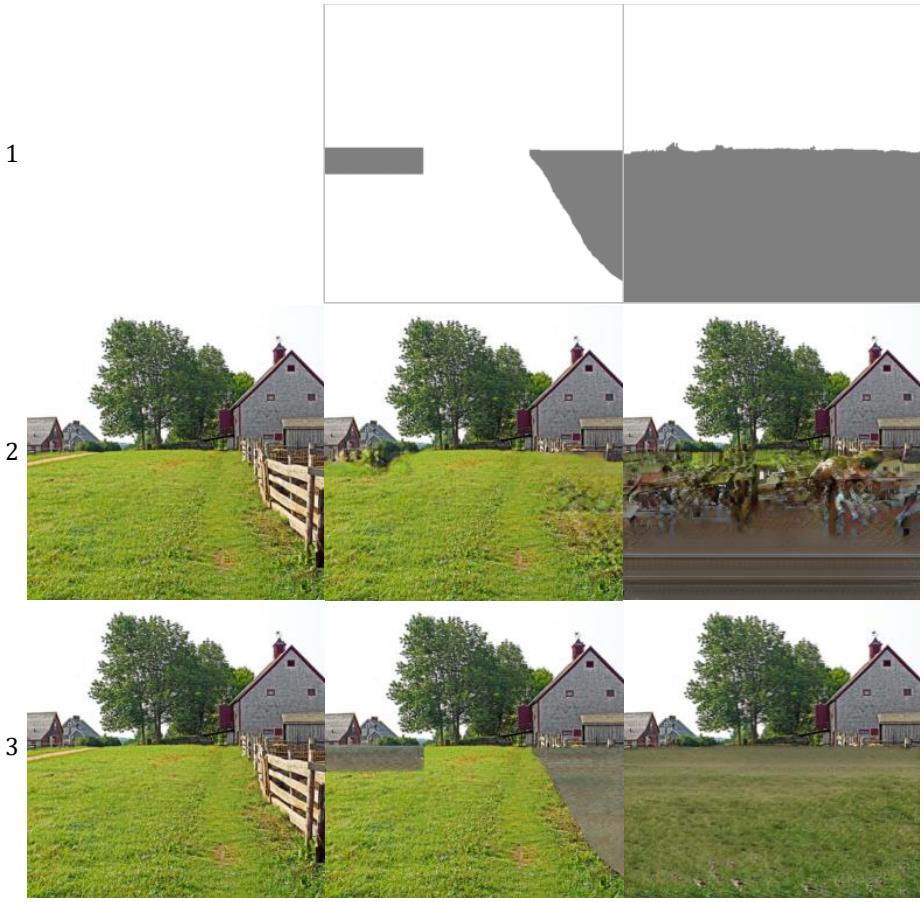

Fig. 16. Comparison with [27] for mask based object removal. Masks are shown at Row 1, where the gray indicates regions to be filled. Column 1 shows the real image. Row 2 and Row 3 show results edited following the given masks by [27] and our method, respectively.