Supplementary Material for Fast Soft Color Segmentation

Naofumi Akimoto[1] Huachun Zhu[2] Yanghua Jin[2] Yoshimitsu Aoki[1]
[1] Keio University [2] Preferred Networks
nakimoto@aoki-medialab.jp {zhu, jinyh}@preferred.jp aoki@elec.keio.ac.jp

1. Network Structures

In Table A and B, we show the architectures of the *alpha predictor* and the *residue predictor*, based on the naming conventions of network components below:

- Conv2d(K, P): 2D convolution with the kernel size of K and the padding of P;
- DeConv2d(K, P): 2D transposed convolution with the kernel size of K and the padding of P;
- BN: Batch normalization.

2. Experimental Settings of Speed Test

When we compare decomposition speed of methods of Aksoy *et al.* [1], Tan *et al.* [3] and ours, Aksoy *et al.* use a palette size of 7, and Tan *et al.* use an palette size with a mean of 6.95 and median of 7. The methods of measurement for each algorithm are presented below.

- We measure the running time as the total time of alpha layer estimation and color layer estimation (not including reading an input image into GPU memory). We execute our Python code on a 3.50 GHz Intel Core i7-7800X CPU and 64GB of RAM and a NVIDIA Quadro P6000 GPU. At each resolution, we average the decomposition time over 20 images as the final results.
- Aksoy *et al.* [1] use parallelized C++ to conduct the experiment.
- The running time reported by Tan *et al.* [3] including the execution time of RGBXY convex hull computation and layer updating. They execute their Python code on a 2.9 GHz Intel Core i5-5257U CPU and 16 GB of RAM. In their method, a layer updating is possible in few milliseconds, but there is no way to bypass the intensive computation of RGBXY convex hull for each image.

3. Qualitative Comparisons

To qualitatively evaluate our method, we compare previous methods with ours on recoloring and decomposition. Figure A shows examples of recoloring, and Figure B and C compare decomposed layers of our method with those of Aksoy *et al.* [1] and Tan *et al.* [3].

References

[1] Yağız Aksoy, Tunç Ozan Aydın, Aljoša Smolić, and Marc Pollefeys. Unmixing-based soft color segmentation for image manipulation. *ACM Trans. Graph.*, 36(2):19:1–19:19, 2017. 1, 3, 4

[2] Huiwen Chang, Ohad Fried, Yiming Liu, Stephen DiVerdi, and Adam Finkelstein. Palette-based photo recoloring. *ACM Transactions on Graphics (TOG)*, 34(4):1–11, 2015. 3

[3] Jianchao Tan, Jose Echevarria, and Yotam Gingold. Efficient palette-based decomposition and recoloring of images via rgbxy-space geometry. *ACM Transactions on Graphics (TOG)*, 37(6):262:1–262:10, Dec. 2018. 1, 3, 4

[4] Jianchao Tan, Jyh-Ming Lien, and Yotam Gingold. Decomposing images into layers via RGB-space geometry. *ACM Transactions on Graphics (TOG)*, 36(1):7:1–7:14, Nov. 2016. 3

Components	Input size	Output size	Output name
Conv2d(3,1), ReLU, BN	H × W × C	(H/2) × (W/2) × (C×2)	Conv-1
Conv2d(3,1), ReLU, BN	(H/2) × (W/2) × (C×2)	(H/4) × (W/4) × (C×4)	Conv-2
Conv2d(3,1), ReLU, BN	(H/4) × (W/4) × (C×4)	(H/8) × (W/8) × (C×8)	-
DeConv2d(3,1), ReLU, BN	(H/8) × (W/8) × (C×8)	(H/4) × (W/4) × (C×4)	Deconv-1
Concatenate(Deconv-1, Conv-2)	-	(H/4) × (W/4) × (C×8)	-
DeConv2d(3,1), ReLU, BN	(H/4) × (W/4) × (C×8)	(H/2) × (W/2)× (C×2)	Deconv-2
Concatenate(Deconv-2, Conv-1)	-	(H/4) × (W/4) × (C×4)	-
DeConv2d(3,1), ReLU, BN	(H/2) × (W/2) × (C×4)	H × W × (C×2)	Deconv-3
Concatenate(Deconv-3, Input image)	-	(H/4) × (W/4) × (C×2+3)	-
Conv2d(3,1), ReLU, BN	H × W × (C×2+3)	H × W × C	-
Conv2d(3,1), Sigmoid	H × W × C	H × W × $\mathrm{C_{out}}$	-

Table A: The network architecture of the alpha predictor that estimates alpha layers from an input image. Specifically, to predict 7 alpha layers, the alpha predictor takes as inputs an image and 7 palette layers of size $H \times W \times 3$ ($1 \times 1 \times 3$ palette colors broadcast across spatial dimensions). Therefore, the total number of input channels is $C = 3 + 7 \times 3$. The output, composed of 7 single-channel alpha layers, has $\mathrm{C_{out}} = 7$ channels.

Components	Input size	Output size	Output name
Conv2d(3,1), ReLU, BN	H × W × C	(H/2) × (W/2) × (C×2)	Conv-1
Conv2d(3,1), ReLU, BN	(H/2) × (W/2) × (C×2)	(H/4) × (W/4) × (C×4)	Conv-2
Conv2d(3,1), ReLU, BN	(H/4) × (W/4) × (C×4)	(H/8) × (W/8) × (C×8)	-
DeConv2d(3,1), ReLU, BN	(H/8) × (W/8) × (C×8)	(H/4) × (W/4) × (C×4)	Deconv-1
Concatenate(Deconv-1, Conv-2)	-	(H/4) × (W/4) × (C×8)	-
DeConv2d(3,1), ReLU, BN	(H/4) × (W/4) × (C×8)	(H/2) × (W/2)× (C×2)	Deconv-2
Concatenate(Deconv-2, Conv-1)	-	(H/4) × (W/4) × (C×4)	-
DeConv2d(3,1), ReLU, BN	(H/2) × (W/2) × (C×4)	H × W × (C×2)	Deconv-3
Concatenate(Deconv-3, Input image)	-	(H/4) × (W/4) × (C×2+3)	-
Conv2d(3,1), ReLU, BN	H × W × (C×2+3)	H × W × C	-
Conv2d(3,1), tanh	H × W × C	H × W × $\mathrm{C_{out}}$	-

Table B: The network architecture of the residue predictor that decomposes an input image into color layers. Specifically, for decomposition into 7 color layers, the residue predictor takes as inputs an image and 7 RGBA palette layers of size $H \times W \times 4$ ($H \times W \times 3$ palette colors stacked on $H \times W \times 1$ processed alpha layers). Therefore, the total number of input channels is $C = 3 + 7 \times 4$. The output, composed of 7 RGB layers, has $\mathrm{C_{out}} = 7 \times 3$ channels.

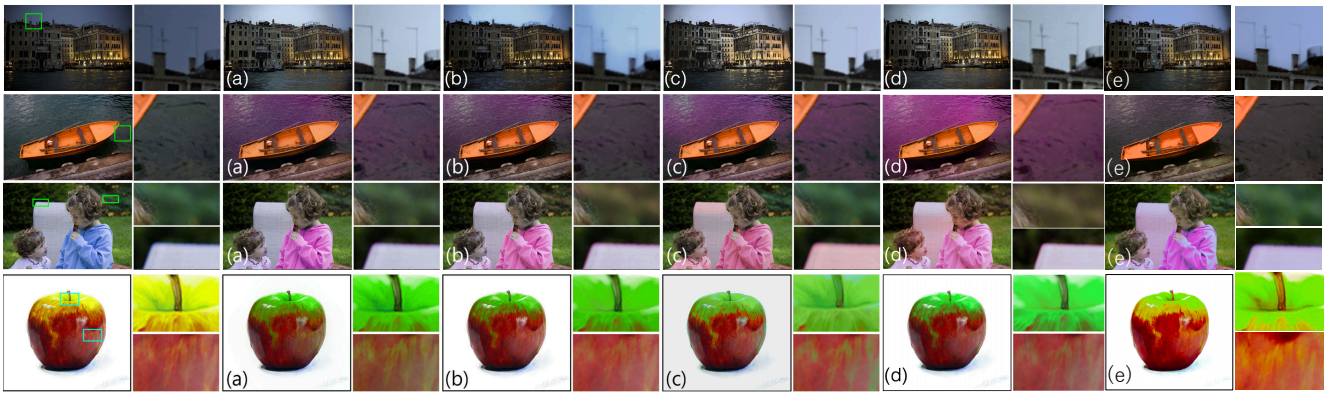

Figure A: Comparisons between previous approaches and our algorithm on recoloring. From left to right: (a) Aksoy *et al.* [1], (b) Tan *et al.* [4], (c) Chang *et al.* [2], (d) Tan *et al.* [3] and (e) our approach. This figure extends the comparisons of recoloring from Tan *et al.* [3].

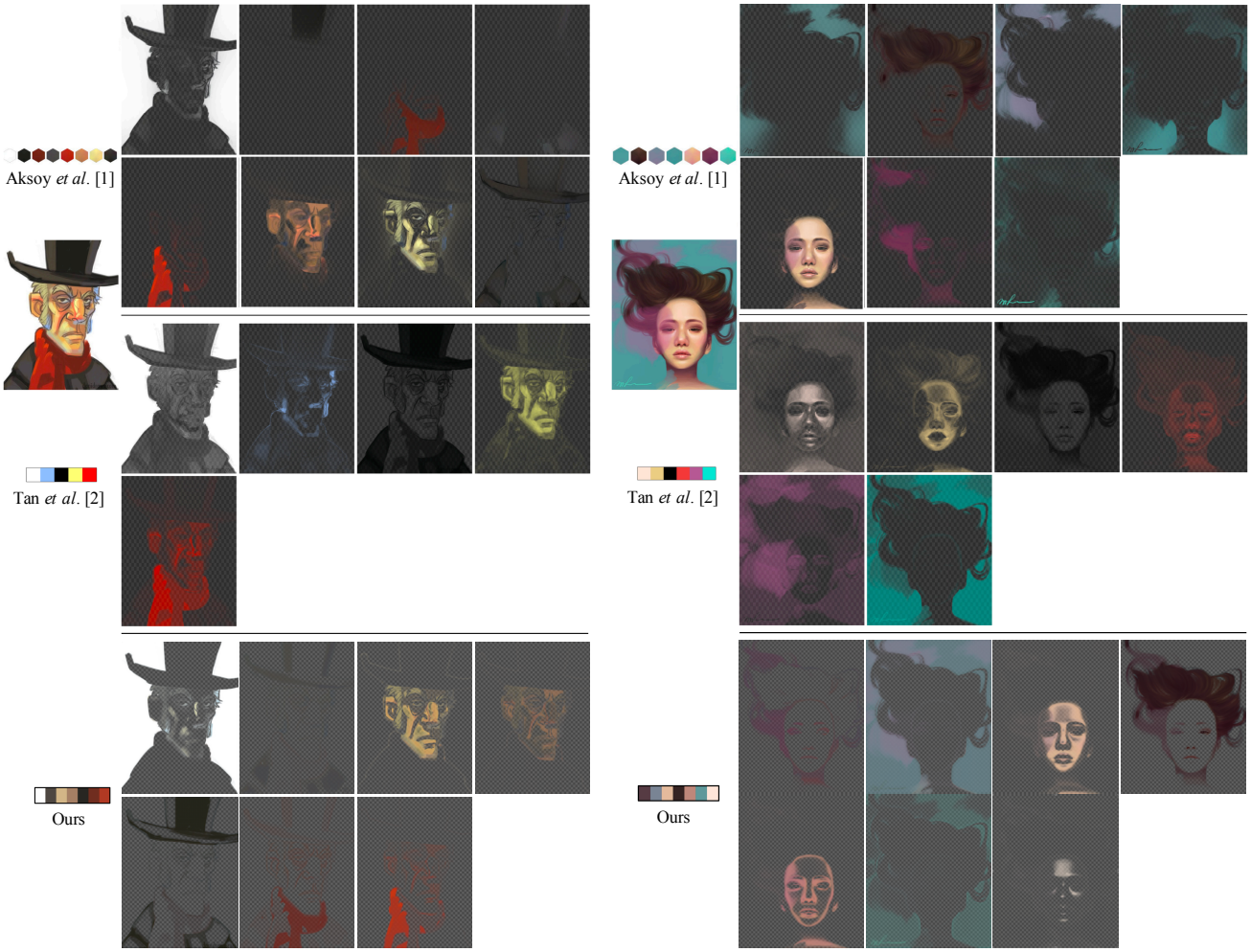

Figure B: Comparisons of decomposed layers between Aksoy *et al.* [1], Tan *et al.* [3], and our approach.

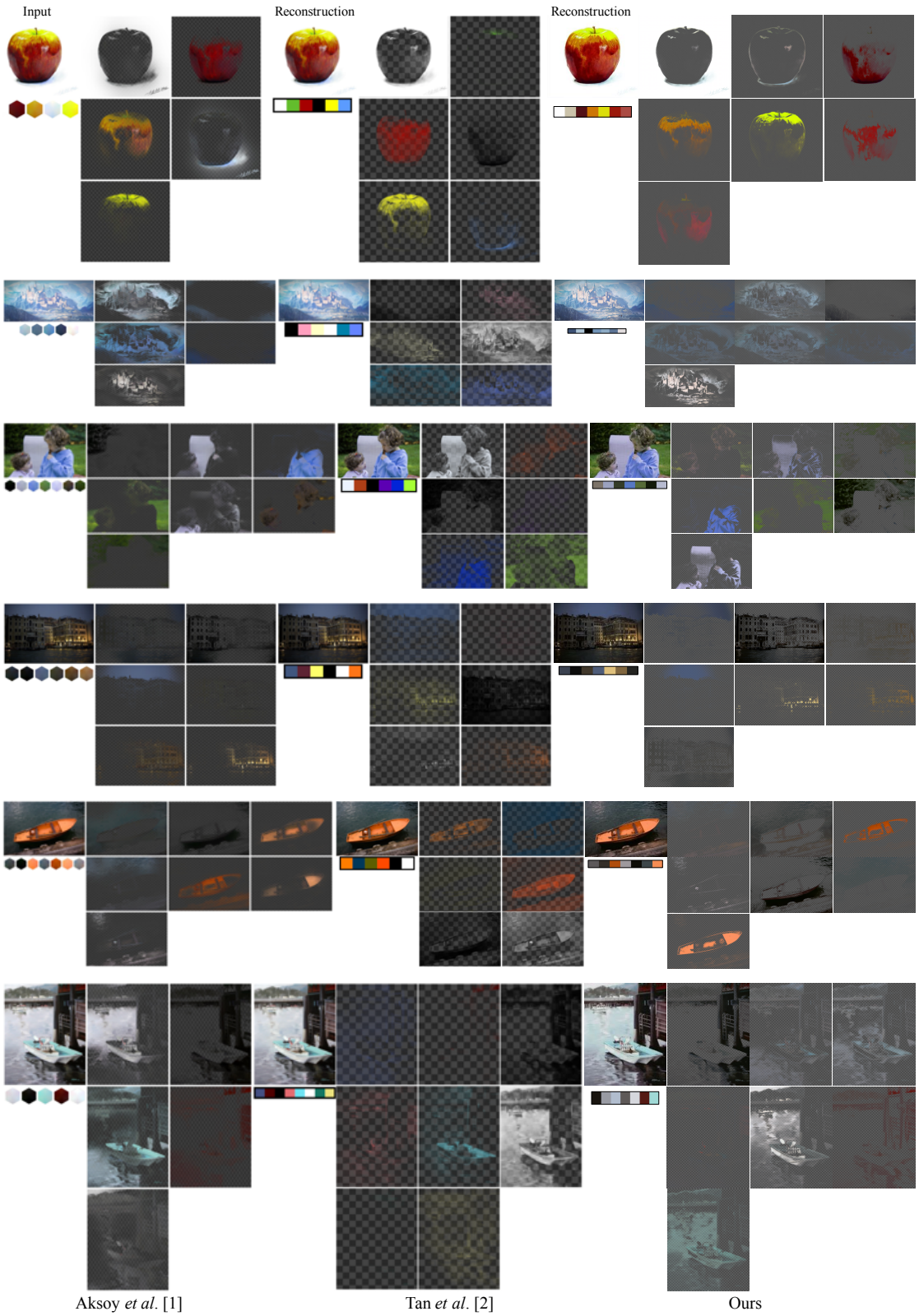

Figure C: More comparisons of decomposed layers between Aksoy *et al.* [1], Tan *et al.* [3], and our approach.