Weakly-Supervised Stitching Network for Real-World Panoramic Image Generation: Supplementary Material

Dae-Young Song[1], Geonsoo Lee[1], HeeKyung Lee[2], Gi-Mun Um[2], and Donghyeon Cho*[1]

[1] Chungnam National University, Daejeon, South Korea
{201501747.o,geonsoo.o,cdh12242}@cnu.ac.kr
[2] Electronics and Telecommunication Research Institute, Daejeon, South Korea
{lhk95,gmum}@etri.re.kr

In this supplementary material, we present additional descriptions of the followings:

- Color pre-processing algorithm.
- The architecture of our stitching network.
- Qualitative results of test images that are not in the main article.

1 Color Processing on Our Dataset

As aforementioned in the main article, the VR camera captures six scenes at the same time. To utilize half of them as the images for weak supervisions, we harmonize their color tone in advance. An overview is described in Figure 1.

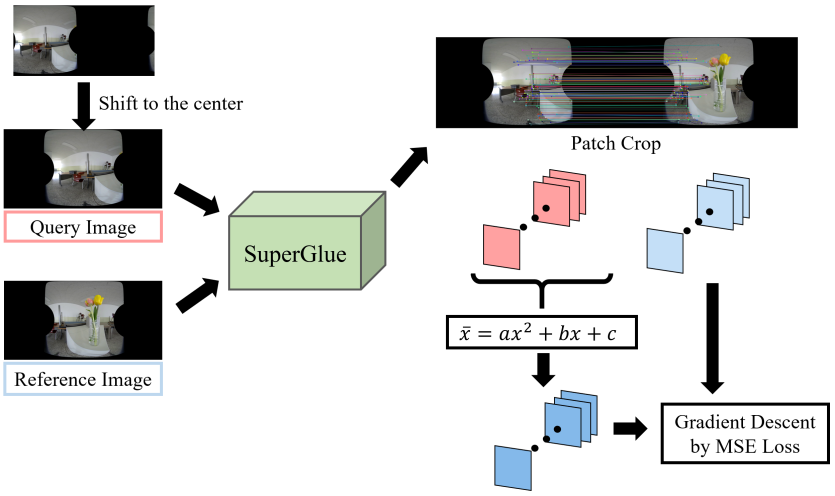

Fig. 1: An overview of polynomial optimization pipeline for color harmonization. After cropping the patches, the polynomial is optimized using the gradient descent method.

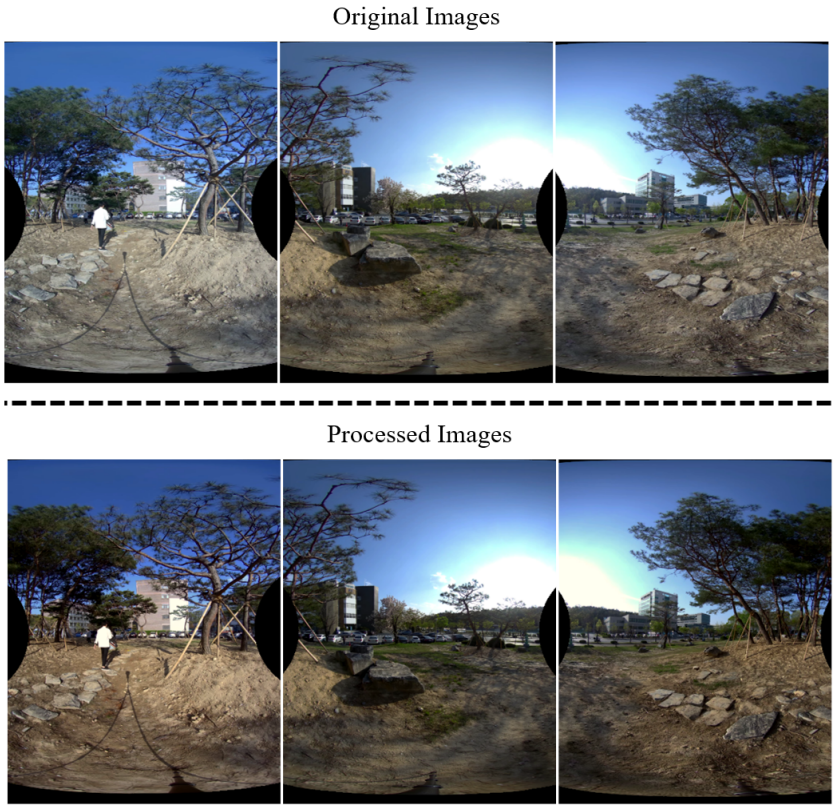

Fig. 2: An example of the color pre-processing result for a set of images for weak supervisions.

Considering that there are few overlapping regions between the images for weak supervisions, input images that transformed the fisheye images into ERP are also used to correct the color, but these images are not used for stitching at all. Note that our stitching network only takes fisheye images that are not corrected as input in the experiments. Since the central area of the images for weak supervisions correspond to the seam line of the input images, our model can learn the harmonized color in this area. As shown in Figure 2, the processed images for weak supervisions have the same color tone. In the pre-processing, we use the outdoor pre-trained SuperGlue weight [3], the learning rate is 0.0008, the patch size is 11, and each pixel is normalized to [0, 1].

2 Detailed Neural Network Description

Our stitching model consists of N encoders that share learnable parameters, a regressor, a shared decoder, and four private decoders. Table 1 is a encoder-decoder description, and they constitute a U-Net-like [2] architecture and have

skip connections. The features from each encoder and the decoder are used for the affine matrix regressor and the private decoders, respectively. A specific description of the regressor and the private decoders is reported in Table 2.

Table 1: The architecture of encoder-decoder in our model, where **k** is the kernel size, **s** the stride, **p** the padding, C_{in} the number of input channels, C_{out} the number of output channels, and **input** corresponds to the input of each layer. Layers marked with a superscript * indicate that they are followed by the batch normalization layer and the ELU [1] activation function. Note that the encoding process repeats N times. "$N\ \otimes$" means that the feature map is concatenated N times at the channel dimension, while $\oplus$ means the skip-connections at the channel dimension.

Encoder						
layer	**k**	**s**	**p**	C_{in}	C_{out}	**input**
dconv1*	7	1	3	3	16	image
dconv1b*	7	2	3	16	16	dconv1*
dconv2*	5	1	2	16	32	dconv1b*
dconv2b*	5	2	2	32	32	dconv2*
dconv3*	3	1	1	32	64	dconv2b*
dconv3b*	3	2	1	64	64	dconv3*
dconv4*	3	1	1	64	128	dconv3b*
dconv4b*	3	2	1	128	128	dconv4*
dconv5*	3	1	1	128	256	dconv4b*
dconv5b*	3	2	1	256	256	dconv5*
dconv6*	3	1	1	256	256	dconv5b*
dconv6b*	3	2	1	256	256	dconv6*
dconv7*	3	1	1	256	512	dconv6b*
dconv7b*	3	2	1	512	512	dconv7*
Decoder						
uconv7*	3	1	1	$512\times N$	$256\times N$	$N\ \otimes$ dconv7b*
idconv7*	3	1	1	$256\times 2N$	$256\times N$	uconv7* $\oplus N\otimes$ dconv6b*
uconv6*	3	1	1	$256\times N$	$256\times N$	idconv7*
idconv6*	3	1	1	$256\times 2N$	$256\times N$	uconv6* $\oplus N\otimes$ dconv5b*
uconv5*	3	1	1	$256\times N$	$128\times N$	idconv6*
idconv5*	3	1	1	$128\times 2N$	$128\times N$	uconv5* $\oplus N\otimes$ dconv4b*
uconv4*	3	1	1	$128\times N$	$64\times N$	idconv5*
idconv4*	3	1	1	$64\times 2N$	$64\times N$	uconv4* $\oplus N\otimes$ dconv3b*
uconv3*	3	1	1	$64\times N$	$32\times N$	idconv4*
idconv3*	3	1	1	$32\times 2N$	$32\times N$	uconv3* $\oplus N\otimes$ dconv2b*
uconv2*	3	1	1	$32\times N$	$16\times N$	idconv3*
idconv2*	3	1	1	$16\times 2N$	$16\times N$	uconv2* $\oplus N\otimes$ dconv1b*
uconv1*	3	1	1	$16\times N$	32	idconv2*
idconv1*	3	1	1	32	16	uconv1*

Table 2: The architecture of shared decoder in our model, where **k** is the kernel size, **s** the stride, **p** the padding, C_{in} the number of input channels, C_{out} the number of output channels, N the number of input images, and **input** corresponds to the input of each layer. "$N \otimes$" means that the feature map is concatenated N times at the channel dimension. Layers marked with a superscript * indicate that they are followed by the batch normalization layer and the ELU activation function.

Regressor						
layer	**k**	**s**	**p**	C_{in}	C_{out}	**input**
conv1*	3	2	1	$512\times N$	512	$N \otimes$ dconv7b*
conv2*	3	1	1	512	512	conv1*
conv3*	3	1	1	512	512	conv2*
conv4*	1	2	1	512	512	conv3*
conv5*	1	1	1	512	512	conv4*
conv6*	1	1	1	512	512	conv5*
avgpool						conv6*
FC				512	$2N\times2\times3$	avgpool
Private Decoders						
Weight Layer						
wconv	3	1	1	16	N	idconv1*
Softmax						wconv
Local Adjustment Layer						
lconv	3	1	1	16	$2N$	idconv1*
Tanh						lconv
Color Correction Layer						
cconv1*	3	1	1	16	16	idconv1*
cconv2*	3	1	1	16	$3N$	cconv1*
cconv3	3	1	1	$3N$	$3N$	cconv2*
Tanh						cconv3

3 Qualitative Comparison

We present more results of test images in Figure 3.

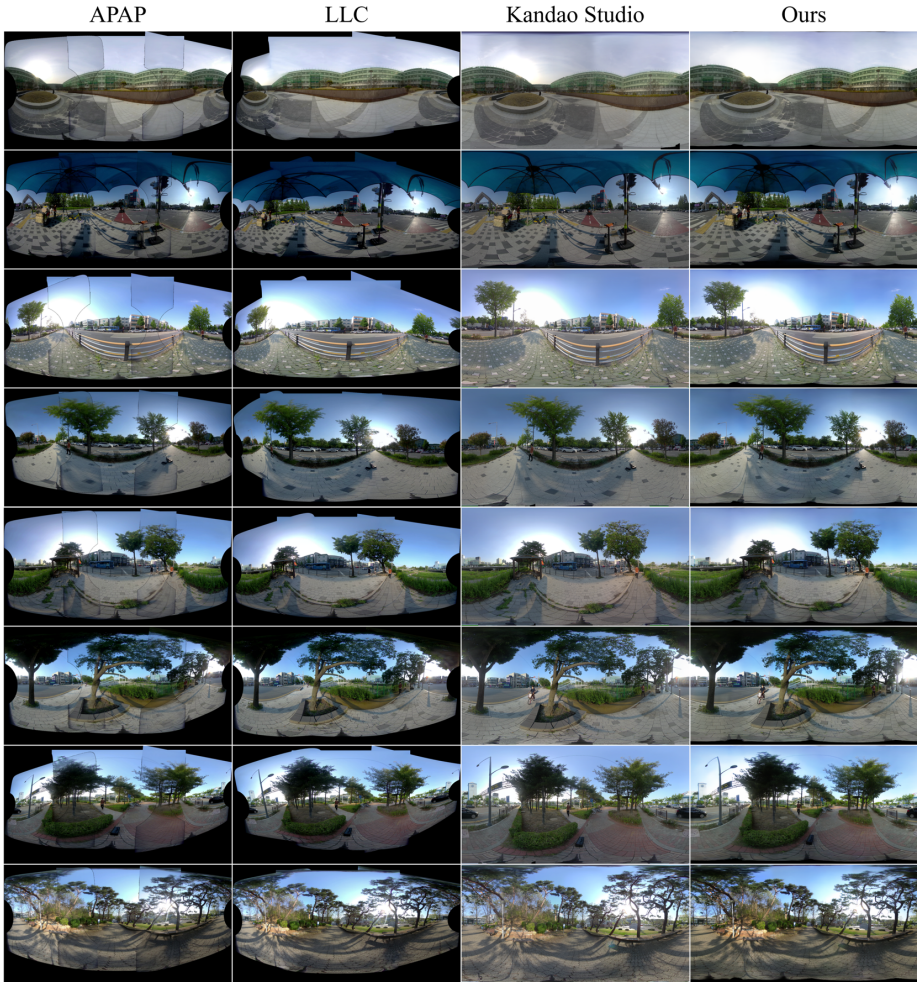

Fig. 3: Qualitative results on our dataset.

References

1. Clevert, D.A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elus). International Conference on Learning Representation (ICLR) (2016) 3
2. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015) 2
3. Sarlin, P.E., DeTone, D., Malisiewicz, T., Rabinovich, A.: SuperGlue: Learning feature matching with graph neural networks. Proc. of Computer Vision and Pattern Recognition (CVPR) (2020), https://github.com/magicleap/SuperGluePretrainedNetwork 2