Supplementary Material: Instance Contour Adjustment via Structure-driven CNN

Shuchen Weng[1], Yi Wei[2], Ming-Ching Chang[3], and Boxin Shi*[1]

[1] NERCVT, School of Computer Science, Peking University
[2] Samsung Research America AI Center
[3] University at Albany, State university of New York
{shuchenweng,shiboxin}@pku.edu.cn
yi.wei1@samsung.com mchang2@albany.edu

In this supplementary material, we show more details about input preparation, model design and training.

1 Structural Cue Completion

The first stage of our method completes the structural cues for the dilated and eroded areas using different approaches because of their respective exclusion rule. Specifically, we propose a diffusion algorithm based on the iterative Gaussian blur operations which under the guidance of the inclusion mask, can propagate the structural cues from the instance area to the corresponding dilated area. Due to the lack of clear correspondences between the eroded area and the external area, the hypothetical distribution of the external instances needs to be inferred for the eroded area. Thus, we modify the structure reconstruction model in [5] to complete the eroded area on the structure image and depth map. To avoid the interference from the instance area, we temporarily cut out the instance which will be pasted back after the completion, and complete the entire instance area as doing the eroded area.

1.1 Preparation of the inclusion mask

The inclusion mask is an important input to our method. An inclusion mask specifies correspondences between the adjusted area and the others. Corresponding areas are set the same label, and visualized in the same color in Fig. 1. In this subsection, we use the "flying pigeon" and "Merlion" in Fig 1 as two examples to illustrate how to prepare an inclusion mask for a dilation case and an erosion case, respectively.

In the inclusion mask of the "flying pigeon" example, the instance area and the dilated areas are both colored in light blue, which indicates that the instance area and the dilated areas are corresponding to each other, and they have no correspondences with the non-instance and non-dilated area which is colored in orange. Therefore, preparing the inclusion mask for a dilation case includes two

* Corresponding author.

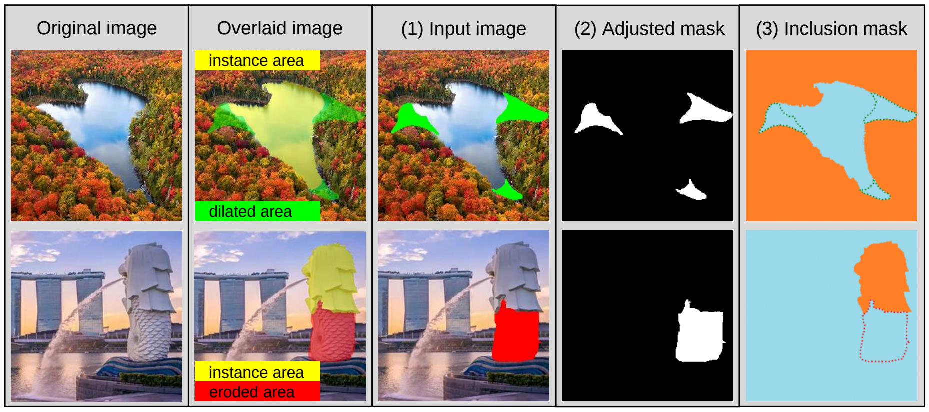

Fig. 1: **The relationship between the inclusion mask and the other inputs.** (2) An adjusted mask is a binary mask which indicates the adjusted area. (3) An inclusion mask specifies correspondences between the adjusted area (enclosed in green/red dotted contour) and the others. Corresponding areas are set the same label, and visualized in the same color.

steps: *(i)* form a union of each instance area and its corresponding dilated areas, and label the union with a unique label (*e.g.*, light blue in the "flying pigeon" example); *(ii)* label the non-instance and non-dilated area with a unique label (*e.g.*, orange in the "flying pigeon" example).

In the inclusion mask of the "Merlion" example, both the eroded area and the non-instance and non-eroded area are colored in light blue, which indicates that the eroded area and the non-instance and non-eroded area are corresponding to each other, and they have no correspondences with the instance area which is colored in orange. Therefore, preparing the inclusion mask for an erosion case also includes two steps: *(i)* form a union of all eroded areas and the non-instance and non-eroded areas, and label the union with a unique label (*e.g.*, light blue in the "flying pigeon" example); *(ii)* label the instance area with a unique label (*e.g.*, orange in the "Merlion" example).

1.2 Diffusion algorithm for the dilated area

To complete the structural cues for the dilated area under the guidance of the inclusion mask, we design a diffusion algorithm based on the iterative Gaussian blur operations. Fig. 2 shows the pipeline of the diffusion algorithm, which is an iterative process with three sub-steps in each iteration. The dilated area is initialized with zeros. The goal of the dilation process is to fill the dilated area with the contexts belonging to the instance area. In order to shield against the artifacts brought by the non-instance area, we split the image plane by the inclusion mask: in Fig. 2, we mark the non-instance area with the brick patterns, which is fixed to zero constantly throughout the whole process.

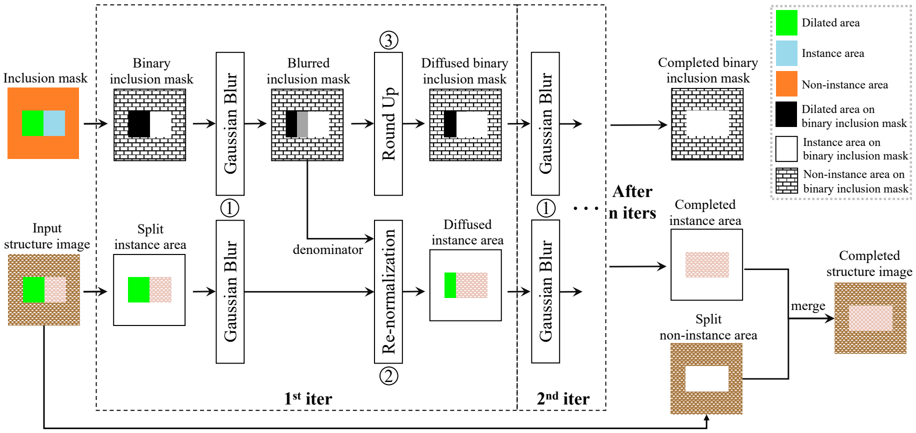

Fig. 2: **The iterative pipeline of the diffusion algorithm of which each iteration consists of three steps (marked with ① ② ③).** The instance area, dilated area and non-instance area are marked in light blue, green and orange, respectively. The white and black in the binary diffusion mask represent values 1 and 0, respectively. The gray in the blurred diffusion mask represents value between 0 and 1. We use the structure image as an example of structural cues in this figure, and the same algorithm is also applied to the depth map.

Then, we aggregate the neighboring contexts belonging to the instance area as the content for the dilated area. The further an instance region is to a dilated region, the lower impact it should have on the dilated region. Thus, we employ a 2D Gaussian kernel of which the weights follow a 2D Gaussian distribution with the center placed at the square kernel center. For each dilated region, we aggregate its contexts by performing the 2D convolution centered on it with the 2D Gaussian kernel, and we name such an operation as “Gaussian Blur”, *i.e.*, Sub-step ① in Fig. 2:

$$g(\mathbf{W},\mathbf{H}) = \frac{1}{\sum_i^K \sum_j^K \mathbf{W}_{i,j}} \sum_i^K \sum_j^K (\mathbf{W} \odot \mathbf{H})_{i,j}, \tag{1}$$

where $\mathbf{W} \in \mathbb{R}_{>0}^{K\times K}$ represents a $K \times K$ Gaussian kernel, with each entry being positive. $\mathbf{H}$ represents a $K \times K$ slice of input, *i.e.*, binary inclusion mask or split instance area. $\odot$ denotes the Hadamard product. $g(\mathbf{W},\mathbf{H})$ represents the Gaussian Blur operation which consists of two sub-operations, *i.e.*, the Hadamard product between $\mathbf{W}$ and $\mathbf{H}$, and the normalization through the division of the Hadamard product by the grand sum of $\mathbf{W}$. We set $K = 15$ in our experiment.

The context aggregation for a specific dilated region should be exclusive to only the instance regions. In (1), the Hadamard product operation complies with this rule because of the zero values in dilated regions. However, the normalization operation violates this rule, because the grand sum considers all regions indiscriminately. Therefore, we need to re-normalize the results of Gaussian Blur by eliminating influences from other dilated regions.

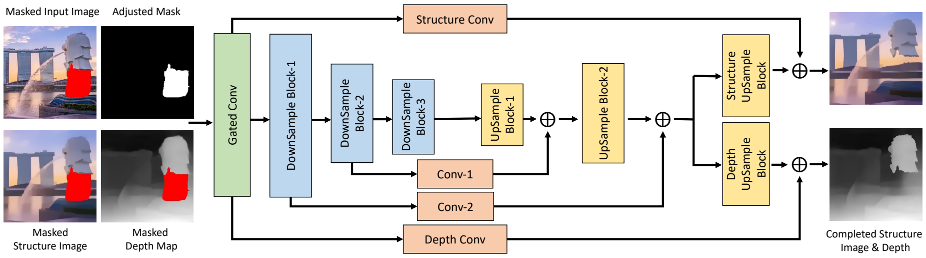

Fig. 3: **The network architecture of structure reconstruction model for the eroded area.**

Thus, we design a re-normalization module, *i.e.*, Sub-step ② in Fig. 2, and we use the blurred inclusion mask output by Gaussian Blur as the denominator for the re-normalization. Let $\mathbf{H}^S$ denote a slice of binary inclusion mask, and we define an index set as $\mathit{\Omega} = \{i, j | \mathbf{H}^S_{i,j} = 1\}$. The blurred inclusion mask can thus be computed by inserting $\mathbf{H}^S$ into (1), removing the expression in the Hadamard product operation involving $\mathbf{H}^S_{i,j} = 0$, and reducing the expression in the Hadamard product operation involving $\mathbf{H}^S_{i,j} = 1$:

$$g(\mathbf{W}, \mathbf{H}^S) = \frac{1}{\sum_i^K \sum_j^K \mathbf{W}_{i,j}} \sum_{i,j \in \mathit{\Omega}} \mathbf{W}_{i,j}. \tag{2}$$

Then, the re-normalized aggregation result h' for a specific dilated region can be computed by having (1) divide by (2).

$$h' = \frac{1}{\sum_{i,j \in \mathit{\Omega}} \mathbf{W}_{i,j}} \sum_i^K \sum_j^K (\mathbf{W} \odot \mathbf{H})_{i,j}. \tag{3}$$

As shown in Fig. 2, these aggregation results constitute the diffused instance area. The blurred inclusion mask can be rounded up to the diffused binary inclusion mask, *i.e.*, Sub-step ③ in Fig. 2. The diffused instance area and diffused binary inclusion mask are fed to the next iteration as inputs. After several iterations, all contents will be completed for the dilated area which can be merged to form the completed structural cue. We apply the same algorithm to the two structural cues, *i.e.*, structure image [8] and depth map.

1.3 Structure reconstruction model for the eroded area

Eroding the instance contour needs to follow the rule that the content of the eroded area should be exclusive to the original instance. To avoid the interference from the original instance, we temporarily cut out the whole instance from image, and hallucinate the content of the whole instance area, the cut-out instance area will be pasted back to the completed structural cues at the end.

Let $\mathbf{I}_{gt}$ denote the ground-truth image, $\mathbf{S}_{gt}$ denote the ground-truth structure image of $\mathbf{I}_{gt}$, and $\mathbf{D}_{gt}$ denote the ground-truth depth map. Let $\mathbf{M}$ denote the

Table 1: Detailed architecture of structure reconstruction model for the eroded area. Please refer to Fig 3 for the network design.

Block's name	Operation's name	Channel	Kernel size	Stride	Activation
	Gated-Conv	8	7	1	ReLU
Downsample Block-1	Gated-Conv	64	4	2	ReLU
	Gated-Conv	128	5	1	ReLU
Downsample Block-2	Gated-Conv	128	4	2	ReLU
	Gated-Conv	256	5	1	ReLU
Downsample Block-3	Gated-Conv	256	4	2	ReLU
Conv-1	Conv	256	5	1	ReLU
Conv-2	Conv	128	5	1	ReLU
Structure Conv	Conv	64	5	1	ReLU
Depth Conv	Conv	64	5	1	ReLU
Upsample Block-1	ResBlock × 4	512	3	1	ReLU
	2×nearest upsample	-	-	-	-
	Gated-Conv	512	5	1	ReLU
Upsample Block-2	ResBlock×4	256	3	1	ReLU
	2×nearest upsample	-	-	-	-
	Gated-Conv	256	5	1	ReLU
Structure Upsample Block	ResBlock×4	128	3	1	ReLU
	2×nearest upsample	-	-	-	-
	Gated-Conv	128	5	1	ReLU
	Conv	128	3	1	Tahn
Depth Upsample Block	ResBlock×4	128	3	1	ReLU
	2×nearest upsample	-	-	-	-
	Gated-Conv	128	5	1	ReLU
	Conv	128	3	1	Tahn

adjusted mask (see Fig. 2 in the paper), *i.e.*, a binary mask which indicates the eroded area. The structure reconstruction model $\mathcal{G}_s$ takes as input the masked image ($\mathbf{I}_{in} = \mathbf{I}_{gt} \odot (1 - \mathbf{M})$), masked structure image ($\mathbf{S}_{in} = \mathbf{S}_{gt} \odot (1 - \mathbf{M})$), masked depth map ($\mathbf{D}_{in} = \mathbf{D}_{gt} \odot (1 - \mathbf{M})$) and adjusted mask $\mathbf{M}$. The outputs of $\mathcal{G}_s$ are the completed structure image $\hat{\mathbf{S}}$ and completed depth map $\hat{\mathbf{D}}$. $\mathcal{G}_s$ is defined as follows:

$$\hat{\mathbf{S}}, \hat{\mathbf{D}} = G_s(\mathbf{I}_{in}, \mathbf{S}_{in}, \mathbf{D}_{in}, \mathbf{M}). \tag{4}$$

The architecture and hyper-parameters of our structure reconstruction model are presented in Fig 3 and Table 1. The training losses include the L1 reconstruction loss $\mathcal{L}_{rec}$ and adversarial loss $\mathcal{L}^{adv}$.

The reconstruction loss is defined as:

$$\mathcal{L}_{rec} = ||\hat{\mathbf{S}} - \mathbf{S}_{gt}||_1 + ||\hat{\mathbf{D}} - \mathbf{D}_{gt}||_1, \tag{5}$$

and the adversarial loss is defined as:

Table 2: Detailed architecture of Structure-driven CNN, which corresponds to Fig. 3 of the paper. “SD-Conv” and “SD-Attn” denote the structure-driven convolution and structure-driven attention proposed in the paper. Above “SD-Attn”, we present the details of the shared encoder. Below “SD-Attn”, we present details of the decoder. θ is specialized hyperparameter of SD-Conv denoting the lower limit of contexts to be sampled.

Operation’s name	Channel	Kernel size (k)	Sample number (θ)	Stride (s)	Activation
SD-Conv	36	7	25	1	ELU
SD-Conv	72	7	9	2	ELU
Gated-Conv	72	3	-	1	ELU
Gated-Conv	144	3	-	2	ELU
SD-Attn					
2× nearest upsample					
Conv	72	1	-	1	ELU
Gated-Conv	72	3	-	1	ELU
2× nearest upsample					
Conv	36	1	-	1	ELU
SD-Conv	36	7	9	1	ELU
Gated-Conv	3	3	-	1	ELU
Gated-Conv	3	3	-	1	Tanh

Table 3: Detailed architecture of the discriminator for Structure-driven CNN.

Operation’s name	Channel	Kernel size (k)	Stride (s)	Activation
Conv	64	5	2	LeakyReLU
Conv	128	5	2	LeakyReLU
Conv	256	5	2	LeakyReLU
Conv	256	5	2	LeakyReLU
Conv	256	5	2	LeakyReLU
Conv	256	5	2	LeakyReLU

$$\begin{aligned}\mathcal{L}_{adv} = &\ \mathbb{E}[\log(1 - D_S(\hat{\mathbf{S}}))] + \mathbb{E}[\log D_S(\mathbf{S}_{gt})] + \\ &\ \mathbb{E}[\log(1 - D_D(\hat{\mathbf{D}}))] + \mathbb{E}[\log D_D(\mathbf{D}_{gt})],\end{aligned} \tag{6}$$

where D_S and D_D are the discriminators for structure image and depth map, both discriminators have the same architecture as those in [5].

2 Structure-driven CNN

In this section, we provide more details about Structure-driven CNN. In Table 2, we present its architecture details which corresponds to Fig. 3 of the paper. We adopt a vanilla convolution network as a discriminator to train the Structure-driven CNN. The detailed architecture of our discriminator is given in Table 3.

We use a L1 reconstruction loss and adversarial loss to train Structure-driven CNN. The total training loss is defined as:

$$\mathcal{L} = \lambda_1 ||\hat{\mathbf{I}} - \mathbf{I}_{gt}||_1 + \mathbb{E}[\log(1 - D_I(\hat{\mathbf{I}}))] + \mathbb{E}[\log D_I(\mathbf{I}_{gt})], \tag{7}$$

where $\hat{\mathbf{I}}$ is the output of Structure-driven CNN, D_I is the discriminator. We set $\lambda_1 = 0.1$ in our experiment.

The training loss is minimized using back-propagation with ADAM optimizer and learning rate 0.0001 and betas $(0.5, 0.999)$.

3 Modifications to Baselines

In order to make a fair comparison with our model, we modify the baselines to take the same five-fold inputs as our method. We introduce the modifications to all baselines one after another below.

PEN-Net [10]. The original PEN-NET [10] only takes the masked image and the adjusted mask as input. We increase the input channel of its first input layer, so that the five-fold inputs, namely, the masked image, the adjusted mask, the inclusion mask, the masked structure image and the masked depth map could be concatenated as one unified input tensor to feed into the network. In our implementation, the inclusion mask is expanded as a 19 channel one-hot tensor (same for the other baselines).

FreeForm [9] has two stages. The first-stage network takes the masked image and the adjusted mask to generate a coarse image, while the second-stage network refines the coarse result output by the first stage. Thus we only modify the input layer of the first-stage network to take the concatenated five-fold inputs tensor but keep the second stage unchanged.

CRFill [11] is also a two-stage network. So we only modify the input of the first-stage network by increasing the input channel of its first layer. The concatenated five-fold inputs tensor instead of only the masked image and the adjusted mask are fed into the network.

Rethink [3] has a encoder-decoder architecture. The encoder takes as input the adjusted mask, the masked image and the masked structure image. In our modification, we concatenate the other two modalities, namely the inclusion mask and the depth map to its original input tensor. We didn't change the decoder and the feature equalization module.

ExtInt [7] decomposes the image inpainting pipeline into two steps by first completing a monochromic image and then progressively colorizing the completed monochrome by learning internally on the masked color image. The first step result can be obtained by any traditional image inpainting model, we adopt the FreeForm [9] as the first-stage network as it achieved the best performance according to the reported results in [7]. To generate the completed monochromic (gray scale) image, we modify the FreeForm [9] to take the five-fold inputs tensor and train the network with the gray scale image as target. The first-stage output is then forwarded to the second-stage model of ExtInt [7] to colorize the adjusted

region. Since the second step is a colorization process, the structural cues could not provide useful information, we keep the second-stage network unchanged.
StFlow [5] is a two-stage model. The first-stage network completes the structure image, while the second-stage network uses the completed structure image to assist image texture generation. We only modify the second-stage network of StFlow [5] by increasing the input channel of the network's first layer. Such that the second-stage network takes as input the adjusted mask, the masked image, the masked depth map, the completed structure image and the inclusion mask.
SPG-Net [6] completes the semantic mask at its first stage and then completes the image contents with the supervision of the completed semantic mask. To allow SPG-Net [6] take the five-fold inputs, we only modify its first stage by increasing its input channel of the first layer. The second stage model remains the same. We also test another version of SPG-Net [6] by feeding the ground-truth semantic mask into its second stage model (the first stage is being short cut). We marked this version with SPG-Net† [6] in § 5 of the paper.
DivStruct [4] is a two-stage inpainting method, where the first stage generates a coarse result, and the second stage uses the generated coarse result to help estimate better attention weights especially for capturing the distant correlations. The original inputs to the first stage are two-fold, *i.e.*, the masked image and the adjusted mask. The original inputs to the second stage are three-fold, *i.e.*, the masked image, the adjusted mask and the coarse result generated by the first stage. We only modify the input layer of the first stage to take the concatenated five-fold inputs tensor but keep the second stage unchanged.
Since **SGE-Net** [2] adopts a pre-trained ResNet [1] as feature extractor for contextual feature learning, the concatenated five-fold inputs could not be applied to its original model due to dimension incompatibility. Therefore, we only report the result of its original model.

In our experiments, we train two versions of the baselines, *i.e.*, the original model and the modified one (marked with *). And report both results in § 5 of the paper.

References

1. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778. IEEE Computer Society (2016) 8
2. Liao, L., Xiao, J., Wang, Z., Lin, C., Satoh, S.: Guidance and evaluation: Semantic-aware image inpainting for mixed scenes. In: ECCV (2020) 8
3. Liu, H., Jiang, B., Song, Y., Huang, W., Yang, C.: Rethinking image inpainting via a mutual encoder-decoder with feature equalizations. In: ECCV (2020) 7
4. Peng, J., Liu, D., Xu, S., Li, H.: Generating diverse structure for image inpainting with hierarchical VQ-VAE. In: CVPR (2021) 8
5. Ren, Y., Yu, X., Zhang, R., Li, T.H., Liu, S., Li, G.: Structureflow: Image inpainting via structure-aware appearance flow. In: ICCV (2019) 1, 6, 8
6. Song, Y., Yang, C., Shen, Y., Wang, P., Huang, Q., Kuo, C.J.: Spg-net: Segmentation prediction and guidance network for image inpainting. In: BMVC (2018) 8
7. Wang, T., Ouyang, H., Chen, Q.: Image inpainting with external-internal learning and monochromic bottleneck. In: CVPR (2021) 7
8. Xu, L., Yan, Q., Xia, Y., Jia, J.: Structure extraction from texture via relative total variation. TOG (2012) 4
9. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Free-form image inpainting with gated convolution. In: ICCV (2019) 7
10. Zeng, Y., Fu, J., Chao, H., Guo, B.: Learning pyramid-context encoder network for high-quality image inpainting. In: CVPR (2019) 7
11. Zeng, Y., Lin, Z., Lu, H., Patel, V.M.: Cr-fill: Generative image inpainting with auxiliary contextual reconstruction. In: ICCV (2021) 7