

Supplementary

In this supplementary document, we provide the network architectures and additional implementation details to complement the main paper. We also present additional quantitative and qualitative comparisons. We will make the source code and pretrained models publicly available to foster future research.

A. Implementation details

A.1 Network architecture

We adopt U-Net [57] as the backbone of the amodal mask prediction net f_{shape} and the composition net f_{comp} . We show the full backbone in Table 5. We use “zero-padding”, batch normalization (“bn”), convolutional transpose (“convt”), and skip connection (“skipk” refers to a skip connection with layer k) in the neural compositing network. The full definition of the neural compositing network is defined as follows.

We modify the CRA model [48] as the backbone of our object content completion net f_{content} . We note that CRA’s architecture is a common paradigm in image inpainting tasks, as similar structures were widely used in [44, 52, 54]. There are, however, several important differences between our object content completion net f_{content} and the CRA model [48]. First, the goal of the two methods is different: our object content completion net f_{content} aims to hallucinate the *invisible regions* of an object from the visible regions. On the other hand, the CRA method [48] targets to recover the entire image (i.e., filling in the missing regions using all the remaining known pixels as contexts). Second, the inputs of the two methods are different. Specifically, the input of our object content completion module, f_{content} , is a triplet consisting of a masked object image ($I_o \odot m_{\text{vis}}$ in the main manuscript), and the two binary masks (m_{vis} and m_{inv} in the main manuscript). The CRA model [48] takes as input a masked image as well as the corresponding mask. Consequently, training data pre-processing for the object completion net f_{content} in our method is different from [48]. In our work, we randomly hide part of the target object to simulate amodal instance completion. In particular, we randomly sample an object masks m_{inv} from the dataset and use them to occlude part of the target object I_o . We apply basic transformations, e.g., scaling and translation, to the sampled object mask m_{inv} such that it has overlap with the target object I_o . Table 2 and Figure 3 in the main manuscript show effectiveness of the modifications on the object completion model f_{content} compared to the original CRA [48]. We use “same” padding and the Exponential Linear Unit (ELU) activation function [9] for all convolution layers. We show the full definition of the object content completion net f_{content} in Table 6-7.

A.2 Training details

We use the Adam optimizer [19] to train the three networks. The initial learning rate is $1e-3$ for the amodal prediction net f_{shape} , $1e-4$ for the object content completion model f_{content} , and $2e-4$ for the neural composition net f_{comp} . We select $\text{beta1} = 0.5$ and $\text{beta2} = 0.9$ for all Adam optimizer [19].

A.3 Inference algorithm

We present the inference procedure in Algorithm 1.

Table 5: The backbone used in the amodal shape prediction net f_{shape} and the neural composition net f_{comp} . *bn*: batch normalization, *convt*: convolutional transpose, *skipk*: a skip connection with layer k .

layers	out channels	stride	activation
4×4 conv	64	2	leaky
4×4 conv, bn	128	2	leaky
4×4 conv, bn	256	2	leaky
4×4 conv, bn	256	2	leaky
4×4 conv, bn	256	2	leaky
4×4 conv, bn	256	1	leaky
4×4 conv, bn	256	1	leaky
skip5, 4×4 convt, bn	256	2	relu
skip4, 4×4 convt, bn	256	2	relu
skip3, 4×4 convt, bn	128	2	relu
skip2, 4×4 convt, bn	64	2	relu
skip1, 4×4 convt, bn	64	2	relu
4×4 conv	4	1	tanh

Table 6: The *coarse network* of the object content completion module f_{content} . *num* refers to the number of layers. *out channels* refers to the number of output channels after the layer. *stride* and *dilation* are the parameters of the convolution operation.

layers	num	out channels	stride	dilation	out shape
5×5 gconv	1	32	2	1	128×128
3×3 gconv	1	64	1	1	128×128
3×3 gconv	1	64	2	1	64×64
3×3 gconv	6	64	1	1	64×64
3×3 gconv	5	64	1	2	64×64
3×3 gconv	4	64	1	4	64×64
3×3 gconv	2	64	1	8	64×64
3×3 gconv	3	64	1	1	64×64
3×3 deconv	1	32	1	1	128×128
3×3 deconv	1	3	1	1	256×256

Table 7: The *refine network* of the object content completion module f_{content} . The notations in the first row are identical to Table 6. *attn* refers to Contextual Attention [52].

layers	num	out channels	stride	dilation	out shape
5×5 gconv	1	32	2	1	128×128
3×3 gconv	1	32	1	1	128×128
3×3 gconv	1	64	2	1	64×64
3×3 gconv	1	128	2	1	32×32
3×3 gconv	2	128	1	1	32×32
3×3 gconv	1	128	1	2	32×32
3×3 gconv	1	128	1	4	32×32
3×3 gconv	1	128	1	8	32×32
3×3 gconv	1	128	1	16	32×32
3×3 gconv + attn	1	128	1	1	32×32
3×3 deconv	1	64	1	1	64×64
3×3 gconv + attn	1	64	1	1	64×64
3×3 deconv	1	32	1	1	128×128
3×3 gconv + attn	1	32	1	1	128×128
3×3 deconv	1	3	1	1	256×256

Algorithm 1 Inference Procedure

Input: an amodal mask prediction model f_{shape} ; an object completion model, denoted as f_{content} ; a neural compositing model, denoted as f_{comp} ; input background I_{bg} and an ordered collection of object images $X = \{x^{(1)}, x^{(2)}, \dots, x^{(M)}\}$, where $x^{(j)} = (I^{(j)}, m_{vis}^{(j)})$, $j \in \{1, 2, \dots, M\}$ and $M = |X|$ (note that $m_{vis}^{(j)}$ is not required to be precise during inference).

- 1: **for** $j = 1, \dots, M$ **do**
- 2: **if** $x^{(j)}$ is occluded **then**
- 3: $\hat{m}_{vis}, \hat{m}_{amodal} = f_{\text{shape}}(x_j, m_{vis})$
- 4: $\hat{I}_o = f_{\text{content}}(I, \hat{m}_{vis}, \hat{m}_{amodal})$
- 5: **else**
- 6: $\hat{I}_o = I \odot m_{vis}$
- 7: **end if**
- 8: $I_{out}, \alpha = f_{\text{comp}}(I_{bg}, \hat{I}_o)$
- 9: $\hat{I} = \alpha \odot I_{out} + (1 - \alpha) \odot I_{bg}$
- 10: **end for**

Return: \hat{I}

B. Ablation study

We conduct an ablation study for the content completion model f_{content} where the new inputs are triplet images consisting of a masked image with the background region preserved ($I \odot m_{inv}$), a visible mask (m_{vis}), and an invisible mask (m_{inv}). In other words, we do *not* hide the background region during training. We train the new content completion model with an identical number of iterations as the prior model. We show the comparison results in Figure 8 which indicates that **removing background regions benefits object reconstruction**.

C. Additional results

C.1 Variance of the comparison for image composition.

We computed Table 3 in the main manuscript by repeating the experiments three times with the object styles transferred towards a randomly selected reference image. Here, we show the variances of the statistical results in Table 8.

Table 8: **Variances of the quantitative comparison for image compositing on the COCOA validation dataset [46].** Three baselines are compared: PB [43], DIB [57], and DovNet [10].

	(0.05, 0.2]			(0.2, 0.4]			(0.4, 0.5]		
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
[43]	1e-2	2e-7	2e-7	5e-2	2e-7	9e-7	1e-1	4e-6	3e-6
[57]	6e-2	3e-6	3e-4	2e-1	3e-5	3e-4	3e-1	2e-4	2e-5
[10]	3e-3	3e-7	2e-8	2e-2	4e-8	5e-7	2e-2	1e-7	2e-6
Ours	7e-2	2e-7	4e-7	1e-2	2e-8	8e-8	8e-2	1e-6	2e-6

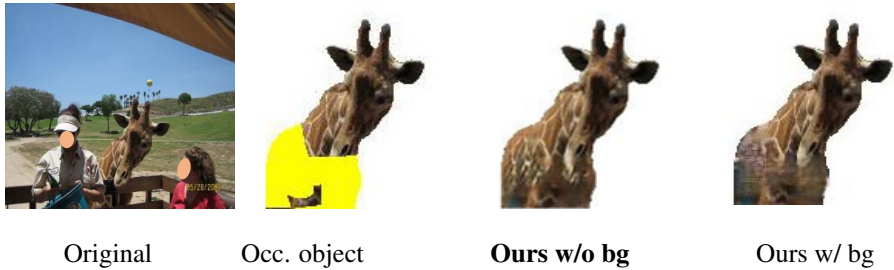


Figure 8: **Ablation study of the object completion model f_{content} .** We predict occluded regions of the objects. *Column 1:* Original images from COCOA validation dataset [59] with visible and amodal masks annotated by [59]; *Column 2:* objects with occluded region marked in yellow and background region in white; *Column 3:* the object completion model with $I \odot m_{\text{vis}}$ as input (i.e., background pixels are empty); *Column 4:* the object completion model with $I \odot (1 - m_{\text{inv}})$ as input (i.e., background pixels are valid).

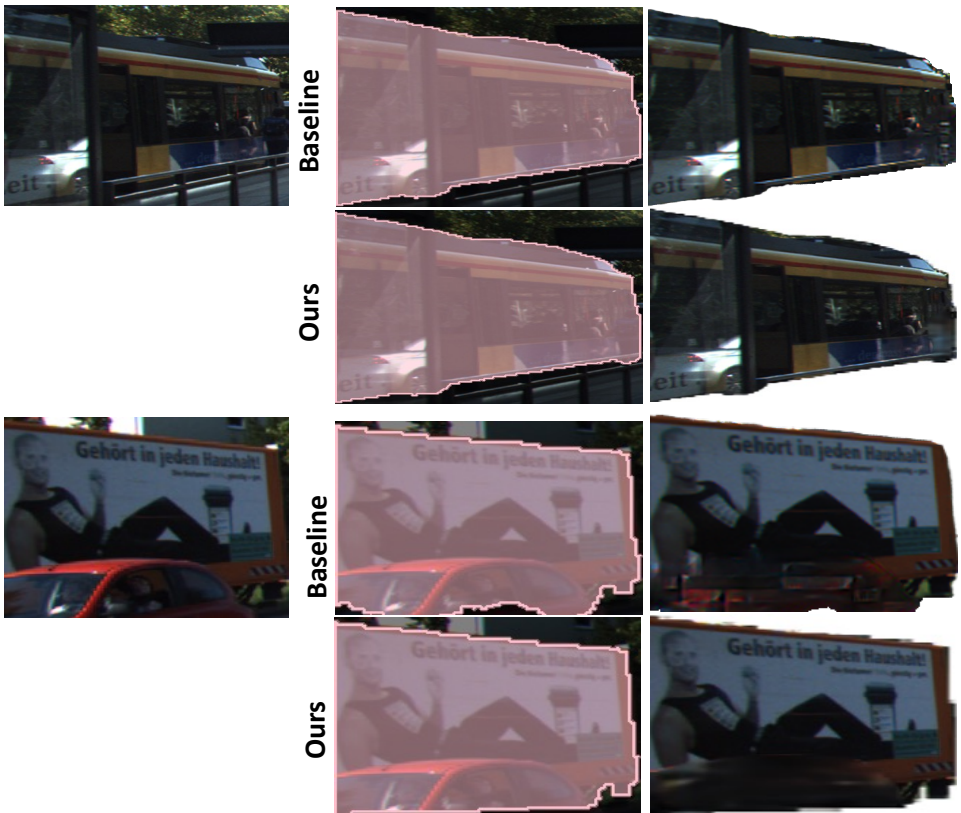


Figure 9: **Results of amodal instance completion on KINS test dataset.** *Column 1:* Original objects, *Column 2:* predicted amodal masks, *Column 3:* synthetic amodal objects.

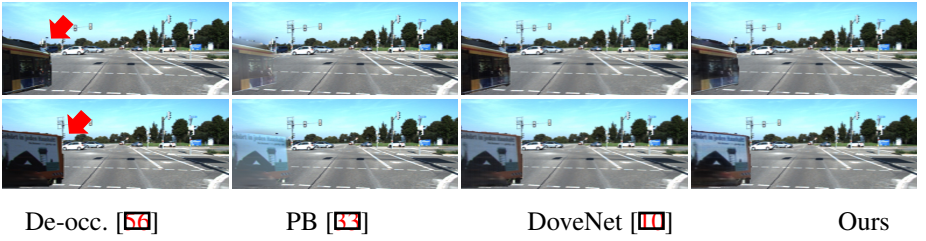


Figure 10: **Results of amodal instance composition on KINS test dataset.** We inserted the instances into the new background image (marked by red arrow). *De-occ.* [56] generated the intact object and then directly pasted the amodal instance into the new background. *PB* and *DoveNet* [10, 53] were used to harmonize into the new background the amodal instance by our amodal mask prediction net f_{shape} and object content completion net f_{content} . *Ours* were achieved by the proposed three modules.

C.2 Amodal instance composition on KINS dataset

We show additional amodal mask prediction and object content completion comparisons in Figure 9 and composition results of the identical instances in Figure 10. Specifically, in Figure 9, we present the predicted amodal masks and the hallucinated results by the baseline method [56] and our approach in column 2-3. Our amodal mask prediction net f_{shape} has a comparable performance with [56], and our content completion net f_{content} can hallucinate the occluded regions of the objects with fewer artifacts (see the occluded corners and the bottom of the vehicles). In Figure 10, we insert the completed objects into a new background image (marked by the red arrow). *De-occ.* [56] does not harmonize the amodal instances with the new background in its applications, leading to unrealistic compositing results. We also notice that *PB* [53] performs unsatisfied when the objects have obvious color contrast with the background. The same issues of *PB* [53] are discussed in prior work [40, 57]. In contrast, our composition net f_{comp} works stably well with fewer artifacts.

C.3 Additional applications

Here we demonstrate several additional applications using our method.

Object re-shuffling. Our object completion model enables re-shuffling objects. Figure 11 shows examples of re-shuffling objects to new locations in the images. A limitation of our model is that the occluded region is smooth, e.g., the blue luggage in Figure 11. We leave this for future improvement.

Object insertion with imperfect inputs. In Figure 12, we show the results of placing two dishes onto an indoor scene. Our method automatically adjusts the foreground instances' colors towards the background images with redundant pixels around objects removed in these cases.



Figure 11: **Object re-shuffling results.** *left:* original images. The arrows indicate object moving directions; *right:* results with objects re-shuffled.

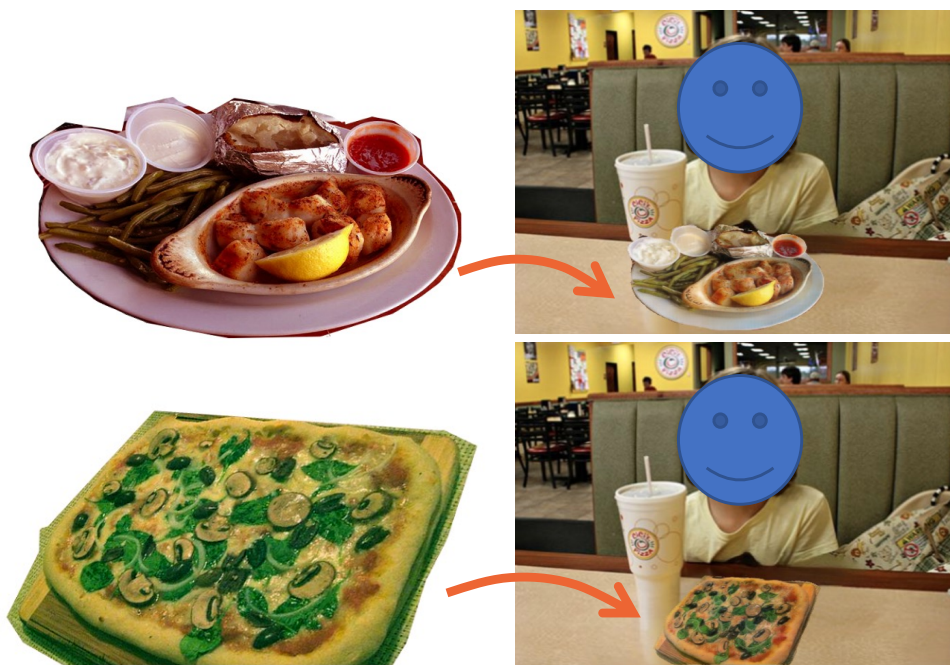


Figure 12: **Object insertion results.** We insert the dishes into the background image. Our method harmonizes the content and refines the imperfect masks.