ARShadowGAN: Shadow Generative Adversarial Network for Augmented Reality in Single Light Scenes

Daquan Liu[1], Chengjiang Long[2]*, Hongpan Zhang[1], Hanning Yu[1], Xinzhi Dong[1], Chunxia Xiao[1,3,4]*
[1]School of Computer Science, Wuhan University
[2]Kitware Inc., Clifton Park, NY, USA
[3]National Engineering Research Center For Multimedia Software, Wuhan University
[4]Institute of Artificial Intelligence, Wuhan University
chengjiang.long@kitware.com, {daquanliu,zhanghp,fishaning,dongxz97,cxxiao}@whu.edu.cn

Abstract

In this supplementary material, we provide more details about the futher motivation clarification, mask generation and 3D models in our Shadow-AR dataset, detailed description about network architecture used in our ARShadowGAN, more ablation studies on input items, visual verification of virtual object masks, as well as more visualization results, to strongly support the main paper. Note that we didn't include all these in the main part of the paper due to the space limit.

1. Motivation

In general, the sense of reality of AR applications is mainly affected by geometry and illumination consistency. The geometry consistency is mainly related to the position and perspective of virtual objects while the illumination consistency is mainly related to the shading and shadow of virtual objects. Since the geometry consistency is more fully studied than that of illumination consistency, producing an image with visual pleasing geometry appearance is not difficult even without full geometry information (such as the marker-based method we use to construct Shadow-AR dataset). However, producing shadows for inserted objects is very challenging under the same conditions. The goal of our work is to achieve direct shadow generation without complex but unreliable inverse rendering techniques for inserted virtual objects in AR images. So the input of our ARShadowGAN contains a synthetic image without virtual shadows and a mask of inserted objects, both of which are very easy to obtain automatically in general AR applications without causing extra annotating.

*This work was co-supervised by Chengjiang Long and Chunxia Xiao.

2. Mask Generation and 3D Models in Our Shadow-AR Dataset

To obtain the virtual object masks in Shadow-AR dataset, we use OpenGL Shading Language (GLSL) to implement the renderer. In fragment shader, we set the virtual object vertex color as white (normalized $color = (1.0, 1.0, 1.0)$) and set the background color as black (normalized $color = (0.0, 0.0, 0.0)$), and re-render the synthetic images with the same configuration used to render synthetic AR images. The masks obtained this way are exactly pixel-wise accurate.

We list all the 13 used 3D models used in Shadow-AR dataset in Figure 1.

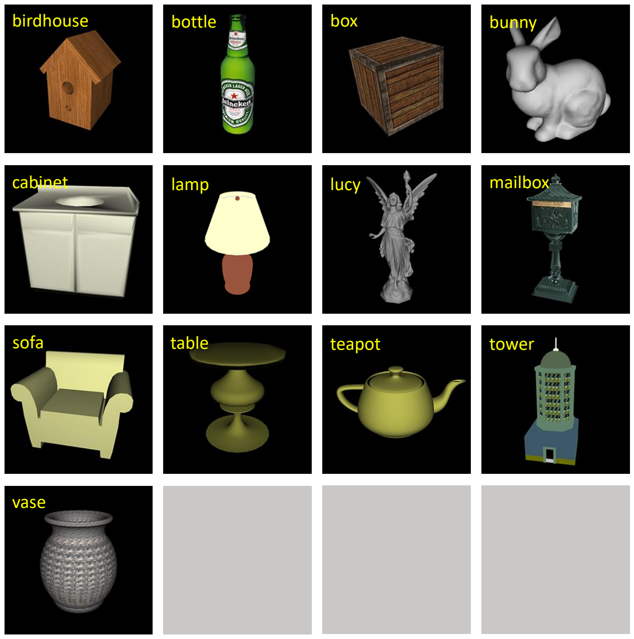

Figure 1. All the 13 used 3D models in our Shadow-AR dataset.

3. Detailed Description about Network Architecture of ARShadowGAN

Detailed network architecture of the discriminator is shown in Table 1. All the convolution operations use valid padding.

Detailed network architecture of the attention block is shown in Table 2. Three skip connections are used: DS(1)-US(4), DS(2)-US(3) and DS(3)-US(2). The output of DS(4) is directly feed to US(1). Note that two decoder branches share the same decoder architecture.

Detailed network architecture of the virtual shadow generator is shown in Table 3. Similarly. Four skip connections are used: DS(1)-US(5), DS(2)-US(4), DS(3)-US(3) and DS(4)-US(2). The output of DS(5) is directly feed to US(1).

Layers	Output Size	Operations
Layer1	$127 \times 127 \times 32$	conv (4×4, stride 2)
Layer2	$62 \times 62 \times 64$	conv (4×4, stride 2)
Layer3	$30 \times 30 \times 128$	conv (4×4, stride 2)
Layer4	$14 \times 14 \times 256$	conv (4×4, stride 2)
Output	$11 \times 11 \times 1$	conv (4×4, stride 1)

Table 1. Architecture of the discriminator.

Layers	Output Size	Operations
DS(1)	$256 \times 256 \times 64$	3-layer res-block
	$128 \times 128 \times 64$	avg pooling (2×2)
DS(2)	$128 \times 128 \times 128$	3-layer res-block
	$64 \times 64 \times 128$	avg pooling (2×2)
DS(3)	$64 \times 64 \times 256$	3-layer res-block
	$32 \times 32 \times 256$	avg pooling (2×2)
DS(4)	$32 \times 32 \times 512$	3-layer res-block
	$16 \times 16 \times 512$	avg pooling (2×2)
US(1)	$32 \times 32 \times 512$	nearest interpolation
	$32 \times 32 \times 512$	dila-conv (3×3, rate 2)
US(2)	$64 \times 64 \times 512$	nearest interpolation
	$64 \times 64 \times 256$	dila-conv (3×3, rate 2)
US(3)	$128 \times 128 \times 256$	nearest interpolation
	$128 \times 128 \times 128$	dila-conv (3×3, rate 2)
US(4)	$256 \times 256 \times 128$	nearest interpolation
	$256 \times 256 \times 64$	dila-conv (3×3, rate 2)
Conv	$256 \times 256 \times 1$	conv (3×3, stride 1)

Table 2. Architecture of the attention block. "dila-conv" is an abbreviation for "dilated conv"

4. More Ablation Studies

To further explore the role of the attention and the input mask, we re-train these ablated versions of our full ARShadowGAN model, in which some input items are removed:

Layers	Output Size	Operations
DS(1)	$256 \times 256 \times 64$	3-layer res-block
	$128 \times 128 \times 64$	avg pooling (2×2)
DS(2)	$128 \times 128 \times 128$	3-layer res-block
	$64 \times 64 \times 128$	avg pooling (2×2)
DS(3)	$64 \times 64 \times 256$	3-layer res-block
	$32 \times 32 \times 256$	avg pooling (2×2)
DS(4)	$32 \times 32 \times 512$	3-layer res-block
	$16 \times 16 \times 512$	avg pooling (2×2)
DS(5)	$16 \times 16 \times 1024$	3-layer res-block
	$8 \times 8 \times 1024$	avg pooling (2×2)
US(1)	$16 \times 16 \times 1024$	nearest interpolation
	$16 \times 16 \times 1024$	conv (3×3, stride 1)
US(2)	$32 \times 32 \times 1024$	nearest interpolation
	$32 \times 32 \times 512$	conv (3×3, stride 1)
US(3)	$64 \times 64 \times 512$	nearest interpolation
	$64 \times 64 \times 256$	conv (3×3, stride 1)
US(4)	$128 \times 128 \times 256$	nearest interpolation
	$128 \times 128 \times 128$	conv (3×3, stride 1)
US(5)	$256 \times 256 \times 128$	nearest interpolation
	$256 \times 256 \times 3$	conv (3×3, stride 1)
Refine	$256 \times 256 \times 64$	composite function
	$256 \times 256 \times 64$	composite function
	$256 \times 256 \times 64$	composite function
	$256 \times 256 \times 64$	composite function
Conv	$256 \times 256 \times 3$	conv (3×3, stride 1)

Table 3. Architecture of the virtual shadow generator.

Models	RMSE	SSIM	S (%)	A (%)	ACC (%)
w/o Attn	7.175	0.962	23.162	21.079	98.446
w/o $\mathcal{A}_{robj}$	7.167	0.963	23.151	21.064	98.449
w/o $\mathcal{A}_{rshadow}$	7.170	0.962	23.156	21.069	98.449
ARShadowGAN	**6.520**	**0.965**	**22.278**	**19.267**	**98.453**
ARShadowGAN-V_D	7.891	0.963	30.846	24.157	97.924
ARShadowGAN-V_G	11.382	0.894	60.325	40.434	84.985

Table 4. Results of more ablation studies. The best scores are highlighted in bold.

- w/o $\mathcal{A}_{robj}$: we use the real shadow attention only.
- w/o $\mathcal{A}_{rshadow}$: we use the real object attention only.
- ARShadowGAN-V_D: the variant in which the final shadow image is the only input for the discriminator.
- ARShadowGAN-V_G: the variant without masks fed into the generator.

We fix the attention block and re-train all these models for the 2^{nd} stage with the same details as our ARShadowGAN. Quantitative results are shown in Table 4. Results of the ablated model without the whole attention block is also listed in the 1^{st} row of Table 4. As we can see, the performance of the model with single attention is almost the same as that without attention. It is partly due to the complementarity of the two attentions. The real object attention and the real shadow attention together promote the virtual shadow inference. ARShadowGAN works better than its variant ARShadowGAN-V_D, which is partly because that three inputs ensure the discriminator to focus more on virtual objects and produced shadows to better optimize the shadow generation. ARShadowGAN outperforms its variant ARShadowGAN-V_G because the mask guides the network directly specify which objects need shadows and avoid extracting background and foreground.

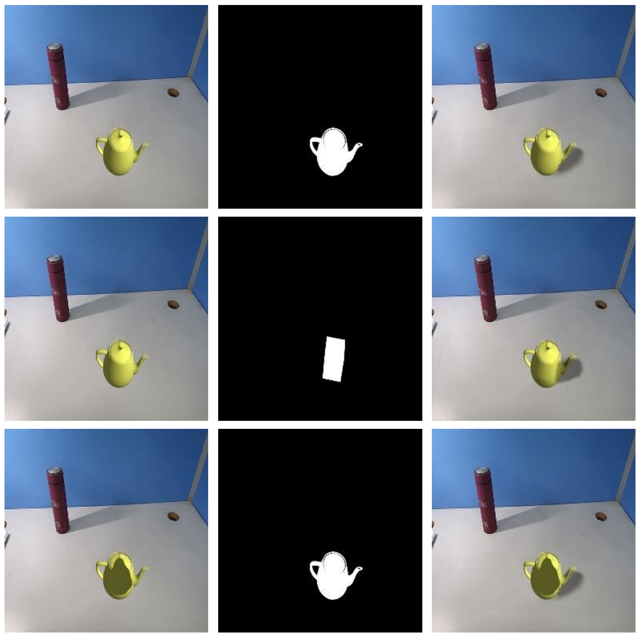
Figure 2. Role of virtual object masks. From left to right are: input images, input masks and output images. In the 2^{nd} row, unmatched mask is input. In the 3^{rd} row, inserted teapot is graffiti.

5. Visual Verification of Virtual Object Masks

We designed one experiment to further verify the role of the mask. As is shown in Figure 2 the 2^{nd} row, we input an image (a teapot inserted) with the unmatched box mask and the produced shadow shape is closer to the box (see the shadow boundaries). In Figure 2 the 3^{rd} row, we graffiti the teapot to eliminate its own shading and ARShadowGAN produces shadow almost same as the 1^{st} row.

Considering the experimental results comprehensively, we can see that the virtual object mask has the following effects:

- Virtual object masks directly specify the inserted virtual objects and guide ARShadowGAN to generate corresponding shadows around virtual objects.
- Virtual object masks mask out the effects of the virtual object shading on produced shadows. The reason is that Shadow-AR dataset does not contain examples in which virtual shadows change with object shading.
- Virtual object masks are related to the shape of the produced shadows.

6. More Visualization Results

We provide more visualization results in Figure 3. Note that the original output attention maps are single-channel grayscale images and we use JET colormap (OpenCV implementation) for visualization. As we can see, ARShadowGAN also works well when there are multiple shadows and occluders, see the 1^{st} and 6^{th} row.

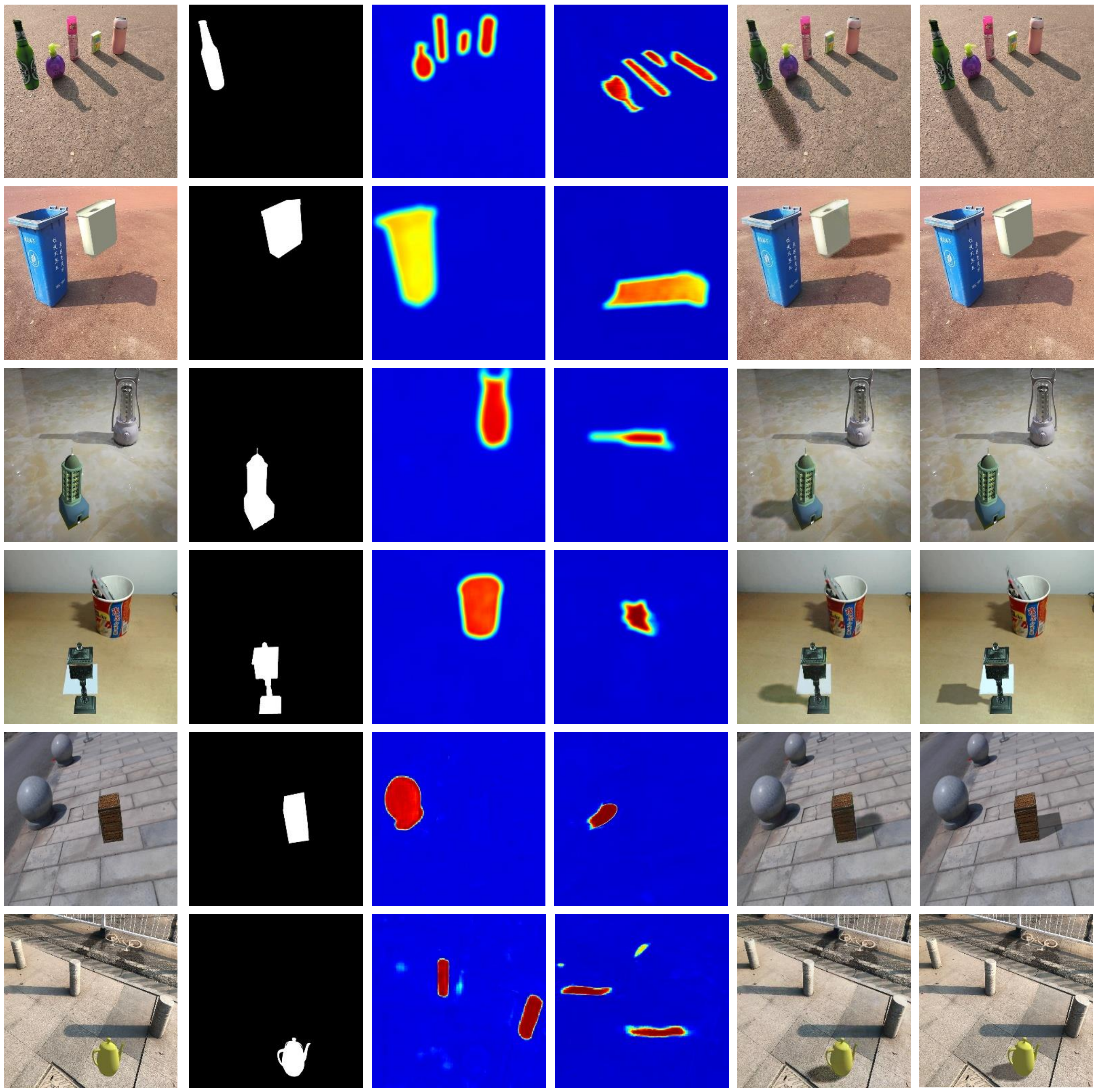

Figure 3. More visualization results. From left to right are: input images without virtual object shadows, input masks, output real object attention maps, output real shadow attention maps, output images with virtual shadows and ground truth.