Seeing Deeply and Bidirectionally: A Deep Learning Approach for Single Image Reflection Removal

Jie Yang*, Dong Gong*, Lingqiao Liu, Qinfeng Shi†

School of Computer Science,
University of Adelaide
{jie.yang01,lingqiao.liu,javen.shi}@adelaide.edu.au,
edgong01@gmail.com

Abstract. Reflections often obstruct the desired scene when taking photos through glass panels. Removing unwanted reflection automatically from the photos is highly desirable. Traditional methods often impose certain priors or assumptions to target particular type(s) of reflection such as shifted double reflection, thus have difficulty to generalize to other types. Very recently a deep learning approach has been proposed. It learns a deep neural network that directly maps a reflection contaminated image to a background (target) image (*i.e.* reflection free image) in an end to end fashion, and outperforms the previous methods. We argue that, to remove reflection truly well, we should estimate the reflection and utilize it to estimate the background image. We propose a cascade deep neural network, which estimates both the background image and the reflection. This significantly improves reflection removal. In the cascade deep network, we use the estimated background image to estimate the reflection, and then use the estimated reflection to estimate the background image, facilitating our idea of seeing deeply and bidirectionally.

1 Introduction

When taking photos through windows or vitrines, reflections of the scene on the same side of the camera, often obstruct the desired scene and ruin the photos. The reflections, however, are often unavoidable due to the limitations on time and/or space. There are practical demands for image reflection removal.

To deal with the image reflection, we first assume that, without the obstruction from the reflection, we can take a clear image, $\mathbf{B} \in \mathbb{R}^{m\times n}$, and then model the reflection contaminated image $\mathbf{I} \in \mathbb{R}^{m\times n}$ as a linear combination of $\mathbf{B}$ and a reflection layer (called reflection) $\mathbf{R} \in \mathbb{R}^{m\times n}$ [1]:

$$\mathbf{I} = \alpha * \mathbf{B} + (1-\alpha) * (\mathbf{K} \otimes \mathbf{R}), \tag{1}$$

where the real scale weight $\alpha \in (0.5, 1)$ is usually assumed as a homogeneous constant [1–3], $\otimes$ is a convolution operator and $\mathbf{K}$ usually represents a Gaussian blurring kernel corresponding a defocus effect on the reflection. Note that $\mathbf{K}$ can also be a delta function (*i.e.* no blur on $\mathbf{R}$) to represent the case where $\mathbf{B}$ and $\mathbf{R}$ are both in-focus.

[*] equal contribution

[†] This work was supported by Australian Research Council grants DP140102270 and DP160100703

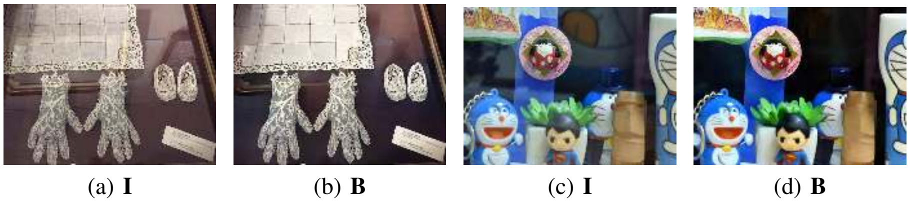

(a) **I** (b) **B** (c) **I** (d) **B**

Fig. 1. An example of single image reflection removal. (a) and (c) are images taken in front of a glass display case, which is degenerated by the reflection. (b) and (d) are the recovered background images of the proposed reflection removal method.

Given an image $\mathbf{I}$ contaminated by reflection $\mathbf{R}$, reflection removal aims to recover the clear background image $\mathbf{B}$. This is challenging since it is highly ill-posed [4]. Some methods thus require multiple images with variations in reflection and/or background as input [1, 5–8] or user assistance to label the potential area of reflection and background [4] to reduce the issue. Multiple images and reliable user guidance are often not easy to acquire, however. To make reflection removal practical, single image reflection removal has received increasing attentions [3, 9, 10].

Solving for $\mathbf{B}$ from a single observation $\mathbf{I}$ usually requires some priors or assumptions to distinguish reflection and background. For example, the ghosting cue [9] is used to identify a special pattern of the shifted double reflection layers from two reflection surfaces. Priors on image gradients are often used to capture the different properties of the different layers [3, 11]. These methods assume the reflection $\mathbf{K} \otimes \mathbf{R}$ is highly blurry due to out-of-focus. Relying on this, recently, a deep learning based method [10] has been proposed to achieve end-to-end single image reflection removal, which utilizes strong edges to identify the background scene, and is trained on the images synthesized with highly blurry reflection layers.

These methods have achieved state-of-the-art performance on many testing examples. However, they also exhibit some limitations in practices such as oversmoothing the image, can not handle the case when the reflections do not have strong blurry or have similar brightness and structure with the background. In this paper, considering the success of the deep learning on image restoration [12–15], we propose to tackle the single image reflection removal by using a cascade deep neural network. Instead of training a network to estimate $\mathbf{B}$ alone from $\mathbf{I}$, we show that estimating not only $\mathbf{B}$, but also the reflection $\mathbf{R}$ (a seemingly unnecessary step), can significantly improve the quality of reflection removal. Since our network is trained to reconstruct the scenes on both sides of the reflection surface (*e.g.* glass pane), and in the cascade we use $\mathbf{B}$ to estimate $\mathbf{R}$, and use $\mathbf{R}$ to estimate $\mathbf{B}$, we call our network bidirectional network (BDN).

2 Related Work

Methods relying on conventional priors Single image reflection removal is a very ill-posed problem. Previous methods rely on certain priors or additional information to handle specific kinds of scenarios.

In some cases, the objects in background layer and reflection layer are approximately in the same focal plane. Some methods exploited gradient sparsity priors to decompose background and reflection with minimal gradients and local features such as edges and corners [16, 17].

In other cases, when taking pictures of objects in the background, the objects reflected from the other side are out of focus due to the different distances to the camera, which leads to the different levels of blur in background and reflection. Li and Brown [3] exploited the relative smoothness and proposed a probabilistic model to regularize the gradients of the two layers. In addition to ℓ_0 gradient sparsity prior, Arvanitopoulos *et al.* [11] proposed to impose a Laplacian data fidelity term to preserve the fine details of the original image. Wan [18] used a multi-scale Depth of Filed map to guide edge classification and used the method in [4] for layer reconstruction afterward.

To distinguish the reflection layer from the background layer, Shih *et al.* [9] studied ghosting cues, which is a specific phenomenon when the glass has a certain thickness and employed a patch-based GMM prior to model the natural image for reflection removal.

Deep learning based methods Some recent works start to employ learning based methods in reflection removal problems.

Fan *et al.* [10] proposed a deep learning based methods to recover background from the image contaminated by reflections. Similar to [3], it also relies on the assumption that the reflection layer is more blurry due to out of focus and they further argue that in some real-world cases, the bright lights contributes a lot to the generation of reflections. They proposed a data generation model to mimic such properties by performing additional operations on the reflection part. They proposed a two-stage framework to first predict an intrinsic edge map to guide the recovery of the background.

Zhang *et al.* [19] used a deep neural network with a combination of perceptual loss, adversarial loss and an exclusion loss to exploit low-level and high-level image information. Wan *et al.* [20] proposed to combine gradient inference and image reconstruction in one unified framework. They also employed perceptual loss to measure the difference between estimation and ground-truth in feature space.

Other related methods Many previous works use multiple observation images as additional information for the recovery of background images. Some use pairs of images in different conditions, such as flash/non-flash [21], different focus [22]. Some use images from different viewpoints, such as video frames [2, 7, 23, 24, 5, 6, 1, 25], through a polarizer at multiple orientations [26, 7, 27], *etc* . But in many real scenarios, we do not have the required multi-frame images for reflection removal. Some work requires manual labelling of edges belonging to reflections to distinguish between reflection and background [4], which is also not suitable for general applications.

3 Proposed method

Focusing on reflection removal, we seek to learn a neural network which is able to recover a reflection-free image from an observation containing reflection obstruction. Specifically, our final goal is to learn a mapping function $\mathcal{F}(\cdot)$ to predict the background image $\widehat{\mathbf{B}} = \mathcal{F}(\mathbf{I})$ from an observed image $\mathbf{I}$. Instead of training only on the image

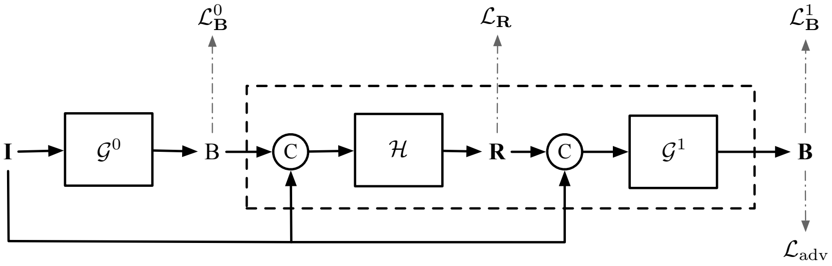

$\mathcal{L}^0_{\mathbf{B}}$
$\mathcal{L}_{\mathbf{R}}$
$\mathcal{L}^1_{\mathbf{B}}$
I
$\mathcal{G}^0$
B
C
$\mathcal{H}$
R
C
$\mathcal{G}^1$
B
$\mathcal{L}_{\text{adv}}$

Fig. 2. Overview of our proposed BDN network architecture and the training objectives. Component C stands for tensor concatenation.

pairs $(\mathbf{I}, \mathbf{B})$'s, we impose the ground truth reflection layers $\mathbf{R}$'s to boost the training of $\mathcal{F}(\cdot)$ by training on a set of triplets $\{(\mathbf{I}_t, \mathbf{B}_t, \mathbf{R}_t)\}_{t=1}^N$. Note that $\mathbf{R}_t$'s are only used in training, not in testing.

3.1 Bidirectional Estimation Model

To directly estimate $\mathbf{B}$ from a given $\mathbf{I}$ in an end-to-end manner, the straightforward idea is to let $\mathcal{F}(\cdot)$ be a neural network taking $\mathbf{I}$ as input and generating $\mathbf{B}$ as output. Our method also includes such a mapping function, and we call it *vanilla generator* $\mathcal{G}^0(\cdot)$. However, our solution further introduces two mapping networks $\mathcal{H}(\cdot)$ and $\mathcal{G}^1(\cdot)$ to estimate the reflection image and refine the background image estimation. In the following parts, we call a composition of $\mathcal{H}$ and $\mathcal{G}^1$ as the bidirectional unit since together they provide estimates for both reflection and background images based on the output of the vanilla generator. The overall structure of the proposed network is shown in Fig. 3.

Vanilla generator The vanilla generator takes the observation $\mathbf{I}$ as the input and generates a background image $\mathbf{B}^0$, *i.e.* $\mathbf{B}^0 = \mathcal{G}^0(\mathbf{I})$, which is the input to the following bidirectional unit.

Bidirectional unit As shown in Fig. 3, the bidirectional unit consists of two components, one for predicting the reflection image and the other for predicting the background image. The first component $\mathcal{H}(\cdot)$ in the bidirectional estimates the reflection image $\mathbf{R}$ from the observation $\mathbf{I}$ and the background estimation $\mathbf{B}^0$ from $\mathcal{G}^0$, *i.e.* $\mathbf{R} = \mathcal{H}(\mathbf{B}^0, \mathbf{I})$. After that, another background estimator $\mathcal{G}^1(\cdot)$ refines the background estimation by utilizing information from the estimation of $\mathbf{R}$ and the original observation $\mathbf{I}$. Thus, the final estimation of background image is calculated by

$$\hat{\mathbf{B}} = \mathcal{G}^1(\mathcal{H}(\mathbf{B}^0, \mathbf{I}), \mathbf{I}). \tag{2}$$

The motivation of using the above bidirectional estimation model is the mutual dependency of the estimation of reflection images and background images. Intuitively, if a good estimation of the reflection image is provided, it will be easier to estimate the background image, vice versa. Also, including the objective of recovering the reflection image provides additional supervision signals to train the network.

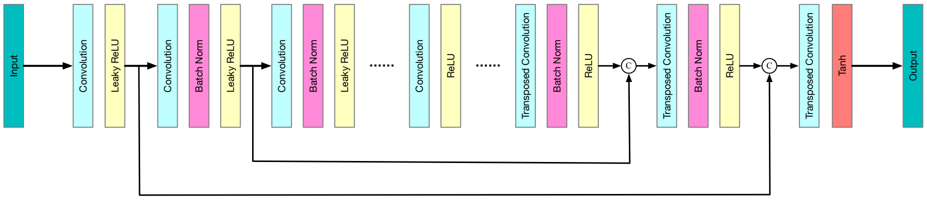


Fig. 3. The network structure of $\mathcal{G}^0$, $\mathcal{H}$ and $\mathcal{G}^1$. C stands for tensor concatenation.

Bidirectional prediction model Based on the above definition of $\mathcal{G}^0(\cdot)$, $\mathbf{H}(\cdot)$ and $\mathcal{G}^1(\cdot)$, we can formulate the whole bidirectional prediction model as:

$$\hat{\mathbf{B}} = \mathcal{G}^1(\mathcal{H}(\mathcal{G}^0(\mathbf{I}), \mathbf{I}), \mathbf{I}), \tag{3}$$

which only takes the observation $\mathbf{I}$ as input. The model shown in Eq. (3) approaches the mapping function $\mathcal{F}(\cdot)$ from the observation $\mathbf{I}$ to the background image $\mathbf{B}$ via a composition of $\mathcal{G}^0(\cdot)$, $\mathcal{H}(\cdot)$ and $\mathcal{G}^1(\cdot)$.

3.2 Network Structure for $\mathcal{G}^0(\cdot)$, $\mathcal{H}(\cdot)$ and $\mathcal{G}^1(\cdot)$

The proposed BDN mainly consists of three subnetworks $\mathcal{G}^0(\cdot)$, $\mathcal{H}(\cdot)$ and $\mathcal{G}^1(\cdot)$. We employ a variation of U-net [28, 29] to implement $\mathcal{G}^0(\cdot)$, $\mathcal{H}(\cdot)$ and $\mathcal{G}^1(\cdot)$. All the three modules share the same network structure (except for the first convolutional layer) but not the same parameters. $\mathcal{G}^0(\cdot)$ has 14 layers, while $\mathcal{H}(\cdot)$ and $\mathcal{G}^1(\cdot)$ has 10 layer. The structure of the network structure is illustrated in Fig. 3.

The U-net employed here contains an encoder part and a decoder part. For the encoder network, all convolution layers are followed by BatchNorm layer [30] and leaky ReLU with slope 0.2, except for the first convolution, which does not have Batch-Norm. For the decoder network, each transposed convolution with stride 2 is used to upsample the feature maps by a factor of 2. The output channel is followed by a Tanh function. All convolutions are followed by a BachNorm layer and a leaky ReLU activation. The kernel size of the filters in all the convolution and transposed convolution layers is fixed to 4×4. The skip connections concatenate each channel from layer i to layer $n - i$ where n is the number of layers. The skip connections combine the information from different layers, specifically allowing low-level information to be shared between input and output. The use of skip connections doubles the number of input channels in the decoder network. The inputs of $\mathcal{H}(\cdot)$ and $\mathcal{G}^1(\cdot)$ are two images. We simply concatenate those two images to make the input have 6 channels rather than 3 color channels.

4 Network Training

4.1 Training Objective

The goal of our network is to learn a mapping function from $\mathbf{I}$ to $\mathbf{B}$ given training samples $\{(\mathbf{I}_t, \mathbf{B}_t, \mathbf{R}_t)\}_{t=1}^{N}$.

Our model consists of three mapping operations: $\mathcal{G}^0 : \mathbf{I} \rightarrow \mathbf{B}$, $\mathcal{H} : (\mathbf{I}, \mathbf{B}) \rightarrow \mathbf{R}$ and $\mathcal{G}^1 : (\mathbf{I}, \mathbf{R}) \rightarrow \mathbf{B}$. Each of the above mapping operations leads to a loss for comparing the compatibility of the estimation and the ground-truth results. In this work, we consider to minimizer the difference between the estimate and the ground truth relying on the ℓ_2-loss and the adversarial loss.

(1) ℓ_2-loss

ℓ_2-loss is widely used to measure the Euclidean distance between the estimated image and the ground-truth image. Minimizing the ℓ_2-loss favors the small mean-squared-error (MSE). Since we have three estimations from the three subnetworks in our network, three respective loss terms are defined and the summation of the three loss term will be used to train the network:

$$\mathcal{L}_2 = \mathcal{L}^0_{\mathbf{B}} + \mathcal{L}_{\mathbf{R}} + \mathcal{L}^1_{\mathbf{B}}, \tag{4}$$

where

$$\mathcal{L}^0_{\mathbf{B}} = \sum_{t=1}^{N} ||\mathcal{G}^0(\mathbf{I}_t) - \mathbf{B}_t||_2, \tag{5}$$

$$\mathcal{L}_{\mathbf{R}} = \sum_{t=1}^{N} ||\mathcal{H}(\mathbf{I}_t, \mathbf{B}) - \mathbf{R}_t||_2, \tag{6}$$

$$\mathcal{L}^1_{\mathbf{B}} = \sum_{t=1}^{N} ||\mathcal{G}^1(\mathbf{I}_t, \mathbf{R}) - \mathbf{B}_t||_2. \tag{7}$$

In (6) and (7), the $\mathbf{B}$ and $\mathbf{R}$ can be the ground truth $\mathbf{B}_t$ or $\mathbf{R}_t$ or the estimates from previous blocks, which depends on the settings in training (See Section 4.2).

(2) Adversarial loss

ℓ_2-loss only calculates the pixel-wise difference between two images, which may not reflect the perceptual difference between two images. Recently, there are an increasing number of works [29, 31, 12, 32, 33] applying the adversarial loss [34] to provide additional supervision for training an image mapping network. The adversarial loss was originally proposed in Generative adversarial networks [34]. The idea is to iteratively train a discriminator to differentiate the ground-truth images from the images generated by a generator at the certain stage of training. Then the objective becomes to encourage the generator to generate images that can confuse the current discriminator. When applying such an adversarial loss to image processing (mapping), we treat the mapping function that maps the observations to the desired output as the generator. The discriminator in the adversarial loss implicitly leans a distribution of the natural images, as an image prior. By applying adversarial loss, the implicit image prior performs as guidance for recovering the images following the natural image distribution. To simplify the training process, we only apply this adversarial loss to the last estimation of the background image, namely, the output of $\mathcal{G}^1$. Formally, the generation function is defined as $\mathcal{F}(\mathbf{I}) = \mathcal{G}^1(\mathcal{H}(\mathbf{B}^0, \mathbf{I})$ and a discriminator $\mathcal{D}$ is trained by optimizing the following objective:

$$\mathcal{L}_{\mathcal{D}} = \sum_{t=1}^{N} \log \mathcal{D}(\mathbf{B}_t) + \sum_{t=1}^{N} \log(1 - \mathcal{D}(\mathcal{F}(\mathbf{I}_t))), \tag{8}$$

and the adversarial loss is defined as

$$\mathcal{L}_{\text{adv}} = \sum_{t=1}^{N} - \log \mathcal{D}(\mathcal{F}(\mathbf{I}_t)) \tag{9}$$

Full objective Finally, we sum the ℓ_2 loss and adversarial loss as the final objective:

$$\mathcal{L} = \mathcal{L}_2 + \lambda \mathcal{L}_{\text{adv}}, \tag{10}$$

where λ is the hyper-parameter that controls the relative importance of the two objectives.

4.2 Training Strategies

Our proposed network has three cascaded modules, the vanilla generator, the reflection estimator and the refined background estimator. These components can be trained independently or jointly. In our work, we explored three ways to conduct training:

- The most straightforward way is to train the whole network end-to-end from scratch.
- Each module can also be trained independently. Specifically, we can progressively train each component until converged and then stack its output to the next component as the input. We call this training strategy as greedy training.
- We can also first train each sub-network progressively and then fine-tune the whole network, which is referred as "greedy training + fine-tuning".

In Section 5.1, we will present the comparison and analysis of these training strategies.

4.3 Implementation

Training data generation We use the model in Eq. (1) to simulate the images with reflections. To synthesize one image, we sample two natural images from the dataset and randomly crop the images into 256×256 patches. One patch is served as background $\mathbf{B}$ and the other is used as reflection $\mathbf{R}$. A Gaussian blur kernel of standard deviation $\sigma \in [0, 2]$ is applied on the reflection patch to simulate the defocus blur may appear on the reflection layer in reality. The two patches are blended using scale weight $\alpha \in [0.6, 0.8]$. The generated dataset contains triplets of $\{(\mathbf{I}_t, \mathbf{B}_t, \mathbf{R}_t)\}_{t=1}^{N}$.

We use images from PASCAL VOC dataset [35] to generate our synthetic data. The dataset contains natural images in a variety of scenes, and it is suitable to represent the scenes where the reflection is likely to occur. We generate 50K training images from the training set of PASCAL VOC dataset, which contains 5717 images.

To compare with [10], which is the only available learning based method as far as we know, we also use the method introduced by [10] to generate another training dataset. It subtracts an adaptively computed value followed by clipping to avoid the brightness overflow when mixing two images. We use the same setting as [10] in data synthesis. The images are also from PASCAL VOC dataset and are cropped at 224×224. The training data is generated from 7643 images, and test set is generated from 850 images.

We trained our network and [10] using both our training data and training data generated by the method of [10].

Training details We implement our model using PyTorch and train the models using Adam optimizer [36] using the default parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, and the initial learning rate is set to be 0.001. Weights are initialized using the method in [37]. The code is available at `https://github.com/yangj1e/bdn-refremv`.

5 Experiments

In this section, we first present comparisons of ablations of our methods to illustrate the significance of our design decisions. Then we quantitatively and qualitatively evaluate our approach on single image reflection removal against previous methods [3, 11, 10] and demonstrate state-of-the-art performance. For numerical analysis, we employed peak-signal-to-noise-ratio (PSNR) and structural similarity index (SSIM) [38] as evaluation metrics.

5.1 Ablation Studies for the Bidirectional Network

Testing data For ablation studies, we use a dataset synthesized from PASCAL VOC [35] validation set, which does not contain any images appeared in the training set. We generate 400 images for testing in ablation studies. The setting of testing data generation is the same as the setting in Sec. 4.3 for training data generation.

To analyze the performance of reflection removal with respect to the scale weight of the background, which reflects relative strength between background and reflection, we generate another smaller dataset. We increment the scale weight from 0.55 to 0.85, with a step size of 0.05 and generate 10 images for each scale weight.

Analysis of the model structure To verify the importance of our bidirectional unit, we compare three model structures: vanilla generator $\mathcal{G}^0$, vanilla generator $\mathcal{G}^0$ + reflection estimator $\mathcal{H}$, and the full bidirectional network (*i.e.* the composition of $\mathcal{G}^0$, $\mathcal{H}$ and $\mathcal{G}^1$, which is referred as $\mathcal{G}^0 + \mathcal{H} + \mathcal{G}^1$ in the following).

All networks are trained from scratch using the settings specified in Sec. 4.3. Since adding the bidirectional unit to vanilla generator will increase the depth of the network and the number of parameters, we cascade three blocks of the vanilla generator to match the depth and number of parameters of our full model. Table 1 shows that merely training a vanilla generator is not good enough to recover reflection free images. Increasing the number of layers of the vanilla generator (see Vanilla $\mathcal{G}^0$ (deep) in Table 1) to enhance the capacity of the model can slightly improve the performance, but it still underperforms our full model. Appending a reflection estimator to vanilla generator improved the performance by regularizing the reconstruction and cascading a background estimator to form a bidirectional unit improve the performance further. Fig. 4 shows several qualitative examples. It can be observed that adding background estimator improved the result of estimation the reflection layer, which enhances the recovery of background in reverse.

Ablation study of the objective functions In Table 1, we compare against ablations of our full loss. To employ adversarial loss, we need to train a discriminator network with

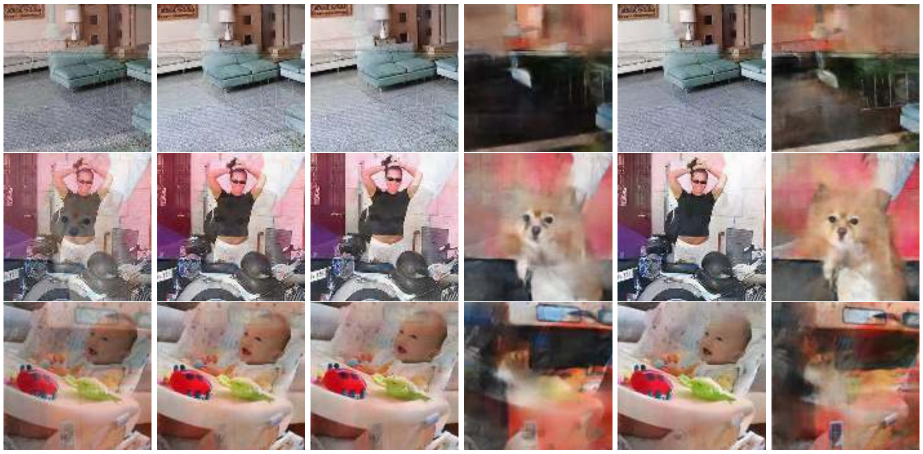


Fig. 4. Visual comparison of our ablation studies on model structure. From left to right: $\mathbf{I}$, $\mathbf{B}$ ($\mathcal{G}^0$), $\mathbf{B}$ ($\mathcal{G}^0$ + $\mathcal{H}$), $\mathbf{R}$ ($\mathcal{G}^0$ + $\mathcal{H}$), $\mathbf{B}$ ($\mathcal{G}^0$ + $\mathcal{H}$ + $\mathcal{G}^1$), $\mathbf{R}$ ($\mathcal{G}^0$ + $\mathcal{H}$ + $\mathcal{G}^1$). Best viewed on screen with zoom.

our model. We adopt the 70×70 PatchGAN of [29] for discriminator, which only penalizes structure at the scale of patches. To train the network with GAN, we pretrain our BDN without adversarial loss first for 2 epochs, and then use the pretrained network to initialize the generator. As the evaluation metrics like PSNR is directly related to MSE, adding adversarial loss has very little improvements compared to directly optimizing ℓ_2 loss in numerical analysis, but for visual appearance, we noticed improvements in restoring the fine details of the background, as shown in Fig. 5.

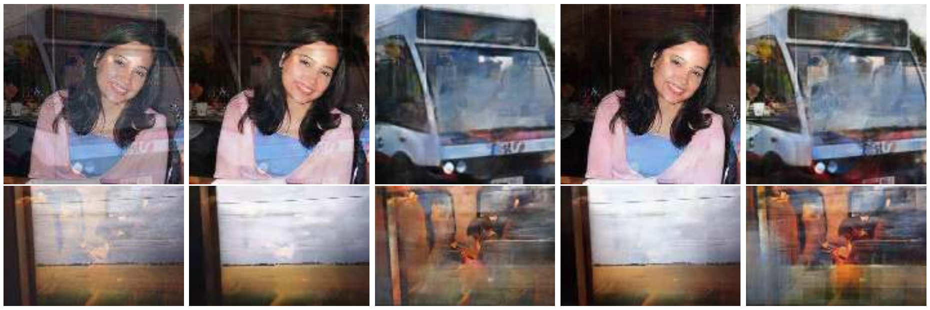


Fig. 5. Visual comparison of our ablation studies on model structure on objective functions. From left to right: $\mathbf{I}$, $\mathbf{B}$ (BDN w/o adversarial loss), $\mathbf{R}$ (BDN w/o adversarial loss), $\mathbf{B}$ (BDN with adversarial loss), $\mathbf{R}$ (BDN with adversarial loss). The upper image is synthetic and the bottom image is real. Best viewed on screen with zoom.

Analysis of training strategy We compare three training strategies specified in Section 4.2. Progressively training each module and then stacking them together, *i.e.* BDN (greedy training + fine-tuning) in Table 1, results in poor performance. The reason is that the reflection estimator and background estimator in the bidirectional unit needs to coordinate, *e.g.* if we train background estimator greedily using the ground truth pairs $\{(\mathbf{I}_t, \mathbf{B}_t)\}_{t=1}^{N}$, but when we stack it after the vanilla generator, the input of this module becomes $\{(\mathbf{I}_t, \widehat{\mathbf{B}}_t)\}_{t=1}^{N}$. Although finetuning from the progressively trained module improves performance and converges quickly, it underperforms end-to-end joint training from scratch, as the greedy initialization is more likely to converge to a bad local optima. For all the following experiments, we train our model from scratch, *i.e.* the three subnetworks are trained jointly.

Table 1. Quantitative comparison with ablation of our methods and with the state-of-the-art methods on 500 synthetic images with reflection generated using the method in Section 4.3, the best results are bold-faced.

	PSNR	SSIM
Vanilla $\mathcal{G}^0$	22.10	0.811
Vanilla $\mathcal{G}^0$ (deep)	22.16	0.817
Vanilla $\mathcal{G}^0$ + $\mathcal{H}$	22.30	0.813
BDN (greedy training)	20.82	0.792
BDN (greedy training + fine-tuning)	22.43	0.825
BDN (joint training, w/o adversarial loss)	23.06	0.833
BDN	**23.11**	**0.835**
Li and Brown [3]	16.46	0.745
Arvanitopoulos *et al.* [11]	19.18	0.760
Fan *et al.* [10]	19.80	0.782

5.2 Quantitative Evaluation

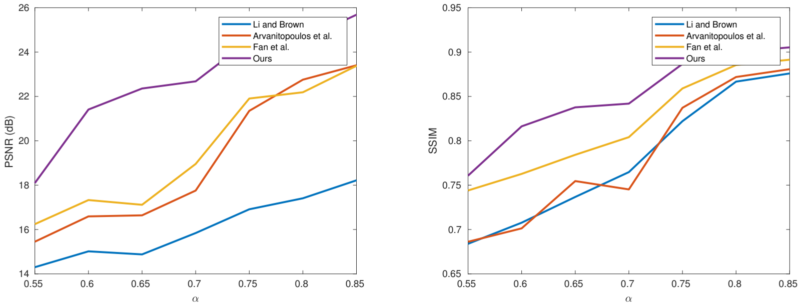
Comparison with the-state-of-the-art We perform quantitative comparison between our method and the-state-of-the-art single image reflection methods of Li and Brown [3], Arvanitopoulos *et al.* [11] and Fan *et al.* [10] using synthetic dataset. The numerical results shown in Table 1 indicates that our method outperforms the state-of-the-art.

Comparison with learning based method We specifically perform some comparisons with [10] as [10] is the only method of solving single image reflection removal problem using deep learning techniques so far. Both [10] and our method require training with synthetic data, but we use different data synthesis mechanism. To compare with [10], we train both our model and [10] using our training data as described in Sec. 4.3 and a training set generated using the algorithm in [10]. Then we evaluate trained models on the corresponding test set, and the results are shown in Table. 2.

Trained on synthetic data in [10], our model achieves comparable performance on the test set in [10] and outperforms [10] when training and testing on our synthetic dataset. Because [10] explicitly utilize edge information and removes reflection by recovering the intrinsic edge of the background image, it relies more on the assumption that the reflection layer is blurry. Therefore, when training in our dataset, which is less

Table 2. Comparison between our method and [10]. Both models are trained and evaluated using the synthetic dataset of [10], the best results are bold-faced.

	Dataset in [10]		Our dataset	
	PSNR	SSIM	PSNR	SSIM
BDN (Ours)	**20.82**	0.832	**23.11**	**0.835**
Fan *et al.* [10]	18.29	**0.8334**	20.03	0.790



(a) PSNR

(b) SSIM

Fig. 6. Evaluation of PSNR and SSIM with the change of scale weight α for the background.

blurry and contains a more general form of reflections, [10] does not perform as well as it does in [10]. By contrast, our model has a stronger capacity to learn from data directly and dealing with less blurry reflections.

Learning based methods train models on synthetic data due to the lack of real labeled data. Since we choose different methods to generate training data and it is difficult to tell which data synthesis method fits the real data the best, we use SIR dataset [39] to evaluate the generational ability of our model on real data with reflections. SIR dataset [39] contains 454 triplets of images shot under various capture settings, *e.g.* glass thickness, aperture size and exposure time, to cover various types of reflections. The dataset contains three scenarios: postcards, solid objects, and wild scenes. The images in this dataset are in size 540×400.

Table 3. Numerical study of the learning based methods on SIR benchmark dataset [39], the best results are bold-faced.

	Postcard		Solid objects		Wild scenes	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Fan *et al.* [10]	**21.0829**	0.8294	**23.5324**	**0.8843**	22.0618	0.8261
BDN (Ours)	20.4076	**0.8548**	22.7076	0.8627	**22.1082**	**0.8327**

Sensitivity to the reflection level Considering the weight α in model (1) reflects the strength of the reflection level, to study the sensitivity of the proposed method to the reflection, we conduct and experiments to evaluate the performance of different methods on the images with different α's. As shown in Fig. 6, with the scale weight of background decreases, it is increasingly difficult to separate reflection from the background. Actually when the background layer and reflection layer have similar brightness and structure, sometimes it is even painful for humans to distinguish them apart. Also, note that the range of α exceeds the range we used in data synthesis, and our methods are robust in different levels of scale weights.

5.3 Qualitative Evaluation

We compare with the previous works using real images collected from previous works [11, 10, 5] and collected from the Internet and wild scenes. Since these images have no ground truth, we can only perform the visual comparison.

Comparison with the method only estimating background Arvanitopoulos *et al.* [11] focus on suppressing the reflections, *i.e.* they do not recover the reflection layer. Therefore, we can only show the comparison with $\mathbf{I}$ and $\mathbf{B}$ in Fig. 7. It can be seen that our method better preserves the details in the background and has fewer artifacts, while [11] tends to oversmooth the image and lose too much information details. For example, in the image of clouds, our result keeps more details of cloud than [11] and in the image of the bag, our result looks more realistic.

Comparison with methods separating two layers We compare our methods with Li and Brown [3], and Fan *et al.* [10], which generate a reflection layer along with the background layer. Although our method focuses on recovering the background rather than separating two layers, our estimation of reflection contains more meaningful information compared to previous methods by looking bidirectional. The quality of the reflection layer reconstructed helps boost our recovery of background in our case. Fig. 8 shows the qualitative comparison results. Our methods outperform the state-of-the-art in recovering the clear background in real scenes with obstructive reflections. Compared to [10], our method better recovers the color of the original image. Because a portion of the light will be reflected back to the side of the background, the objects in the background usually look pale compared to the observation directly without glass. This is reflected by the scale operation when generating our training data.

In Fig. 9, we show an examples of failure case. The image, which is from [39], is taken using two postcards through a thick glass. The reflection is very strong and contains ghosting artefacts, while the background is very blurry, and the interactions between reflections have very complex structure. None of the methods works well in this case.

6 Conclusion

In this paper, we studied the single image reflection removal problem. Motivated by an idea that one can estimate the reflection and use it to boost the estimation of the background, we propose a deep neural network with a cascade structure for single image

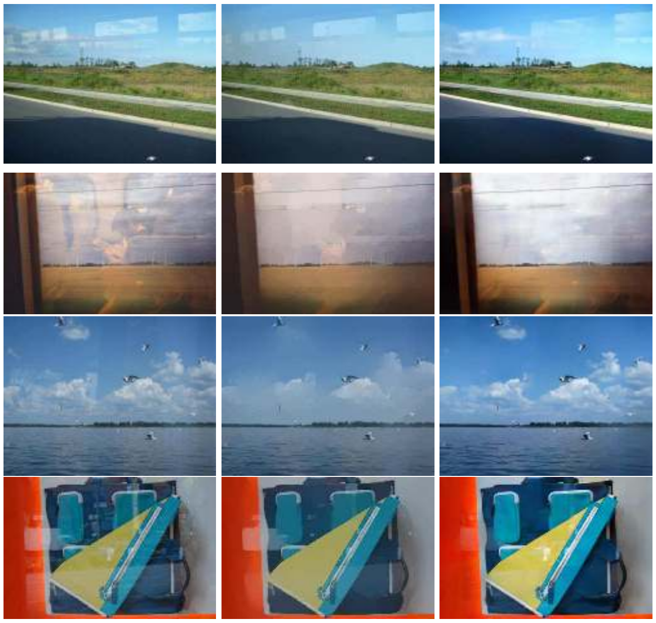


Fig. 7. Comparison with the method of Arvanitopoulos *et al.* [11] on real images. From left to right: **I**, **B** ([11]), **B** (Ours). [11] tends to be oversmooth and our results look more natural. Best viewed on screen with zoom.

removal, which is referred as the bidirectional network (BDN). Benefiting from the powerful supervision, the proposed BDN can recover the background image effectively. Extensive experiments on synthetic data and the real-world data demonstrate that the proposed methods work well in diverse scenarios.

References

1. Xue, T., Rubinstein, M., Liu, C., Freeman, W.T.: A computational approach for obstruction-free photography. ACM Transactions on Graphics **34**(4) (2015) 79
2. Szeliski, R., Avidan, S., Anandan, P.: Layer extraction from multiple images containing reflections and transparency. In: CVPR. Volume 1., IEEE (2000) 246–253
3. Li, Y., Brown, M.S.: Single image layer separation using relative smoothness. In: CVPR, IEEE (2014) 2752–2759
4. Levin, A., Weiss, Y.: User assisted separation of reflections from a single image using a sparsity prior. IEEE Trans. on PAMI **29**(9) (2007)
5. Li, Y., Brown, M.S.: Exploiting reflection change for automatic reflection removal. In: ICCV. (2013) 2432–2439
6. Guo, X., Cao, X., Ma, Y.: Robust separation of reflection from multiple images. In: CVPR, IEEE (2014) 2187–2194
7. Sarel, B., Irani, M.: Separating transparent layers through layer information exchange. In: ECCV. (2004) 328–341

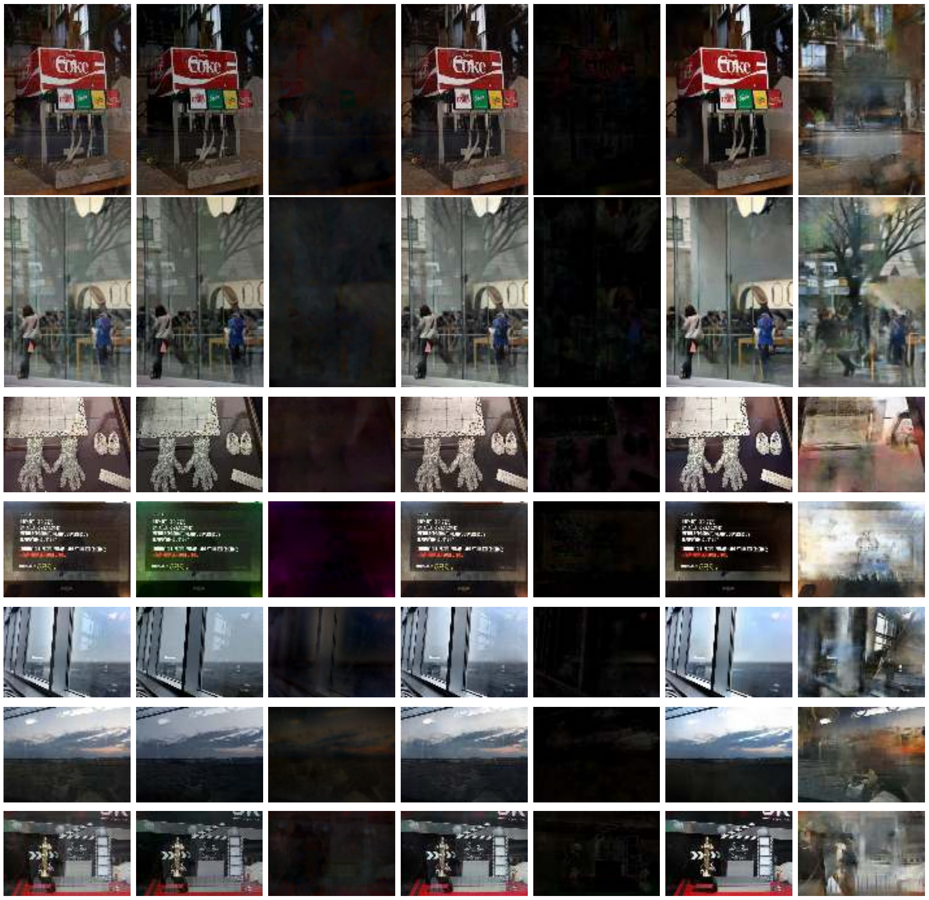


Fig. 8. Comparision of our method with the-stat-of-the-art on real images. From left to right: $\mathbf{I}$, $\mathbf{B}$ ([3]), $\mathbf{R}$ ([3]), $\mathbf{B}$ ([10]), $\mathbf{R}$ ([10]), $\mathbf{B}$ (Ours), $\mathbf{R}$ (Ours). Our networks has clearer background estimation and better color recovery. Best viewed on screen with zoom.

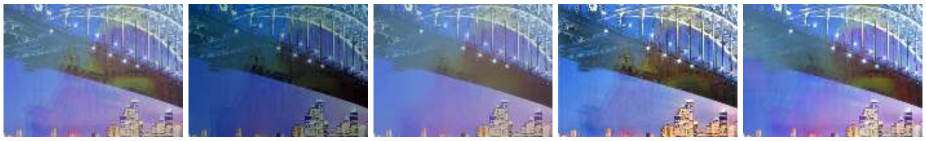


Fig. 9. An example of failure case. From left to right: $\mathbf{I}$, $\mathbf{B}$ ([3]), $\mathbf{B}$ ([11], $\mathbf{B}$ ([10]), $\mathbf{B}$ (Ours)

8. Han, B.J., Sim, J.Y.: Reflection removal using low-rank matrix completion. In: CVPR. Volume 2., IEEE (2017)

9. Shih, Y., Krishnan, D., Durand, F., Freeman, W.T.: Reflection removal using ghosting cues. In: CVPR, IEEE (2015) 3193–3201

10. Fan, Q., Yang, J., Hua, G., Chen, B., Wipf, D.P.: A generic deep architecture for single image reflection removal and image smoothing. In: ICCV. (2017) 3258–3267
11. Arvanitopoulos, N., Achanta, R., Süsstrunk, S.: Single image reflection suppression. In: CVPR, IEEE (2017) 1752–1760
12. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A.P., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network. In: CVPR. Volume 2., IEEE (2017) 4
13. Gong, D., Yang, J., Liu, L., Zhang, Y., Reid, I., Shen, C., van den Hengel, A., Shi, Q.: From motion blur to motion flow: a deep learning solution for removing heterogeneous motion blur. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2017)
14. Mao, X., Shen, C., Yang, Y.B.: Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In: Advances in neural information processing systems. (2016) 2802–2810
15. Gong, D., Zhang, Z., Shi, Q., Hengel, A.v.d., Shen, C., Zhang, Y.: Learning an optimizer for image deconvolution. arXiv preprint arXiv:1804.03368 (2018)
16. Levin, A., Zomet, A., Weiss, Y.: Learning to perceive transparency from the statistics of natural scenes. In: NIPS. (2003) 1271–1278
17. Levin, A., Zomet, A., Weiss, Y.: Separating reflections from a single image using local features. In: CVPR. Volume 1., IEEE (2004) 306–313
18. Wan, R., Shi, B., Hwee, T.A., Kot, A.C.: Depth of field guided reflection removal. In: ICIP, IEEE (2016) 21–25
19. Zhang, X., Ng, R., Chen, Q.: Single image reflection separation with perceptual losses. In: CVPR, IEEE (2018)
20. Wan, R., Shi, B., Duan, L.Y., Tan, A.H., Kot, A.C.: Crrn: Multi-scale guided concurrent reflection removal network. In: CVPR, IEEE (2018) 4777–4785
21. Agrawal, A., Raskar, R., Nayar, S.K., Li, Y.: Removing photography artifacts using gradient projection and flash-exposure sampling. ACM Transactions on Graphics **24**(3) (2005) 828–835
22. Schechner, Y.Y., Kiryati, N., Basri, R.: Separation of transparent layers using focus. IJCV **39**(1) (2000) 25–39
23. Gai, K., Shi, Z., Zhang, C.: Blind separation of superimposed moving images using image statistics. IEEE Trans. on PAMI **34**(1) (2012) 19–32
24. Sinha, S.N., Kopf, J., Goesele, M., Scharstein, D., Szeliski, R.: Image-based rendering for scenes with reflections. ACM Transactions on Graphics **31**(4) (2012) 100–1
25. Yang, J., Li, H., Dai, Y., Tan, R.T.: Robust optical flow estimation of double-layer images under transparency or reflection. In: CVPR. (2016) 1410–1419
26. Schechner, Y.Y., Shamir, J., Kiryati, N.: Polarization and statistical analysis of scenes containing a semireflector. JOSA A **17**(2) (2000) 276–284
27. Kong, N., Tai, Y.W., Shin, J.S.: A physically-based approach to reflection separation: from physical modeling to constrained optimization. IEEE Trans. on PAMI **36**(2) (2014) 209–221
28. 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, Springer (2015) 234–241
29. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: CVPR, IEEE (2017) 5967–5976
30. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML. (2015) 448–456
31. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: ICCV, IEEE (2017) 2242–2251
32. Lettry, L., Vanhoey, K., van Gool, L.: Darn: A deep adversarial residual network for intrinsic image decomposition. In: WACV, IEEE (2018) 1359–1367

33. Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., Webb, R.: Learning from simulated and unsupervised images through adversarial training. In: CVPR. Volume 2., IEEE (2017) 5
34. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems. (2014) 2672–2680
35. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. IJCV **88**(2) (2010) 303–338
36. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
37. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: ICCV. (2015) 1026–1034
38. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. on Image Processing **13**(4) (2004) 600–612
39. Wan, R., Shi, B., Duan, L.Y., Tan, A.H., Kot, A.C.: Benchmarking single-image reflection removal algorithms. In: ICCV, IEEE (2017) 3942–3950