CEILNet: a Cascaded Edge and Image Learning Network

(the first to solve the challenging layer- separation problem of reflection removal from single images using deep learning techniques)

## Article

Fan, Q., Yang, J., Hua, G., Chen, B., & Wipf, D. (2017). A generic deep architecture for single image reflection removal and image smoothing. In Proceedings of the IEEE International Conference on Computer Vision (pp. 3238-3247).

## Key

**Main:**

Propose a deep neural network structure that exploits edge information in addressing representative low- level vision tasks.(layer separation and image filtering)

**Problems and Methods:**

1. single image reflection removal (layer separation):
   - a mild reflection smoothness assumption using neural network
   - a novel synthetic data generation method that acts as a type of weak supervision to generate enough trainnig data
2. image smoothing (image filtering)

**Background:**

The old methods are simply infeasible in many practical situations.

**Why edge based solutions:**

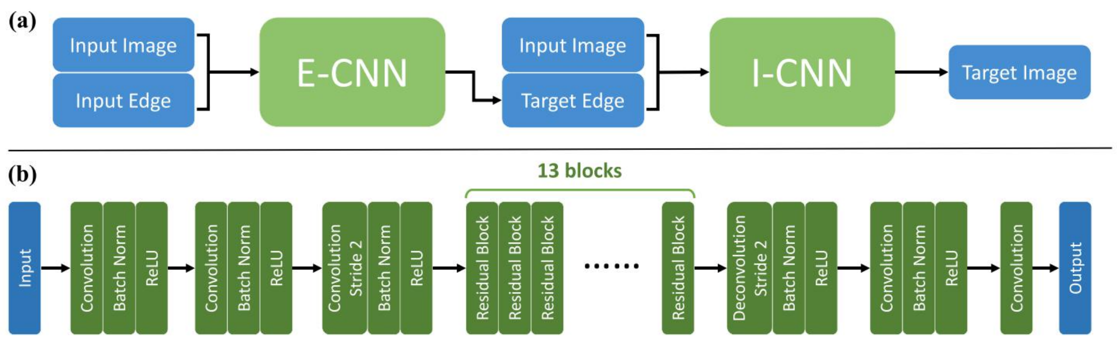
One exploitable property in the reflection removal problem is that the gradients or perceptual structures of the two layers exhibit different distributions, since reflections often display a greater degree of blurring

## Data

### 一、Network Structure

**Steps:**

1. predicting the edge maps of the target images via a deeply supervised sub-network
2. reconstructing the target images by leveraging the predicted edge maps




**E-CNN(get the edges of target image instead of the input image):**

- A more effective edge representation instead of binary edge map: the mean absolute color difference between a center pixel and its four-connected neighbors to represent more information in the edge map

$$\mathbf{E}_{x,y} = \frac{1}{4}\sum_{c}\left(|\mathbf{I}_{x,y,c} - \mathbf{I}_{x-1,y,c}| + |\mathbf{I}_{x,y,c} - \mathbf{I}_{x+1,y,c}| + |\mathbf{I}_{x,y,c} - \mathbf{I}_{x,y-1,c}| + |\mathbf{I}_{x,y,c} - \mathbf{I}_{x,y+1,c}|\right) \quad (1)$$

where $x, y$ are the pixel coordinates and $c$ refers to the channels in the RGB color space.

- The source image with its edge map as an additional channel for input: the edge map of the source image is a part of the image which also underlines the edge features of the source image

**I-CNN:**

- The input image and the target edge are combined to be a 4-channel tensor as input, similar to E-CNN

### 二、Training

1. Train E-CNN and I-CNN in parallel, with loss functions of Eq. 4 and Eq. 5 respectively.
2. Jointly train (fine-tune) E-CNN and I-CNN end-to-end, with loss in Eq. 6.

**Loss:**

- E-CNN:

$$l_E(\theta) = ||\mathbf{E}^t - \mathbf{E}^{t*}||_2^2. \quad (4)$$

- I-CNN:

$$l_I(\theta) = \alpha\,||\mathbf{I}^t - \mathbf{I}^{t*}||_2^2 + \beta\left(||\nabla_x \mathbf{I}^t - \nabla_x \mathbf{I}^{t*}||_1 + ||\nabla_y \mathbf{I}^t - \nabla_y \mathbf{I}^{t*}||_1\right). \quad (5)$$

- CEILNet:

$$l(\theta) = l_I(\theta) + \gamma\, l_E(\theta). \quad (6)$$

**Data Generation:**

Assumption: Reflection is blurry relative to the background layer but can have large intensity

Randomly pick two natural images normalized to $[0,1]$ as background $\mathbf{B}$ and reflection $\mathbf{R}$ respectively, then:

1. $\tilde{\mathbf{R}} \leftarrow gauss\_blur_\sigma(\mathbf{R})$ with $\sigma \sim \mathcal{U}(2,5)$
2. $\mathbf{I} \leftarrow \mathbf{B} + \tilde{\mathbf{R}}$
3. $m \leftarrow mean(\{\mathbf{I}(\mathbf{x},c)\,|\,\mathbf{I}(\mathbf{x},c) > 1, \forall \mathbf{x}, \forall c{=}1,2,3\})$
4. $\tilde{\mathbf{R}}(\mathbf{x},c) \leftarrow \tilde{\mathbf{R}}(\mathbf{x},c) - \gamma\cdot(m-1), \forall \mathbf{x}, \forall c$; $\gamma$ set as 1.3
5. $\tilde{\mathbf{R}} \leftarrow clip_{[0,1]}(\tilde{\mathbf{R}})$
6. $\mathbf{I} \leftarrow clip_{[0,1]}(\mathbf{B} + \tilde{\mathbf{R}})$

Output $\mathbf{I}$ as the synthesized image with $\mathbf{B}$ as the ground-truth background layer.

### 三、Data

Table 1. Result comparison for the image smoothing task (learning an $L_0$ filter [37]). CEILNet outperformed Domain Transform (DT) [10] and simple I-CNNs without E-CNN by large margins.

| | MSE | PSNR | SSIM |
|---|---|---|---|
| DT + input image edge | 124.41 | 27.38 | 0.806 |
| DT + pred. edge by E-CNN | 51.26 | 31.17 | 0.964 |
| DT + GT edge | 45.67 | 31.66 | 0.971 |
| I-CNN only | 37.79 | 32.58 | 0.969 |
| I-CNN only (64 layers) | 31.86 | 33.33 | 0.973 |
| I-CNN with input edge (64 layers) | 22.50 | 34.86 | 0.979 |
| CEILNet | **13.34** | **37.10** | **0.989** |

Table 2. Quantitative comparison of our method with Li and Brown [23] on 100 synthetic images with reflection.

| PSNR | | SSIM | |
|---|---|---|---|
| [23] | Ours | [23] | Ours |
| 15.50 | **18.55** | 0.786 | **0.857** |

## Comments

**TODO:**

1. color attenuation issue observed in deep networks?

--another two papers to read **'Accurate image super-resolution using very deep convolutional networks'** and **'Deeply-recursive convolutional network for image super-resolution'.**

**Idea:**

1. Use both input image and edge map as the whole input which underline the edge information.

2. First get the target edge map then get target image.

3. Training each network separately and then Train both of them together.

4. Exploit a new way to synthesize the data based on the principle of glass reflection.

5. Take advantage of a single structure for two different tasks.

## Why

Learn the background of the research

<table>
<tr><th colspan="5">Reflection separation with three special losses</th></tr>
<tr><th>Article</th><th>Key</th><th>Data</th><th>Comments</th><th>Why</th></tr>
<tr>
<td>Zhang, X., Ng, R., & Chen, Q. (2018). Single image reflection separation with perceptual losses. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4786-4794).</td>
<td>Main:<br>Create a fully convolutional network trained end-to-end with losses that exploit low-level and high-level image information to solve the problem of separating reflection from a single image<br>Problems and Methods:<br>single image reflection removal (layer separation):<br>Three novel loss functions<br>Background:<br>The old methods are simply infeasible in many practical situations.</td>
<td>

一、Network Structure<br>
Input: 1472(hypercolumn features from VGG 19) + 3 dimensions (RGB) --><br>
1*1 convolution to reduce feature to 64 dimensions --><br>
8 layers 3*3 dilated convolutions(The dilation rate varies from 1 to 128) --><br>
Output: two images(background and reflection) concatenated in 6 dimensions<br>
<br>
二、Training<br>
1. Feature loss<br>
Here, we compute the feature loss by feeding the predicted image layer and the ground truth through a pre-trained VGG-19 network $\Phi$. We compute the $L^1$ difference between $\Phi(f_T(I;\theta)$ and $\Phi(T)$ in selected feature layers:<br>

$$L_{\text{feat}}(\theta) = \sum_{(I,T)\in\mathcal{D}} \sum_{l} \lambda_l \|\Phi_l(T) - \Phi_l(f_T(I;\theta))\|_1, \quad (2)$$

where $\Phi_l$ indicates the layer $l$ in the VGG-19 network. The weights $\{\lambda_l\}$ are used to balance different terms in the loss function. We select the layers 'conv1_2', 'conv2_2', 'conv3_2', 'conv4_2', and 'conv5_2' in the VGG-19 network.<br>
<br>
2. Adversarial loss<br>
Loss for the discriminator $D$ is:<br>

$$\sum_{(I,T)\in\mathcal{D}} \log D(I, f_T(I;\theta)) - \log D(I,T), \quad (3)$$

where $D(I,x)$ outputs the probability that $x$ is a natural transmission image given the input image $I$. Then our adversarial loss is:<br>

$$L_{\text{adv}}(\theta) = \sum_{I\in\mathcal{D}} -\log D(I, f_T(I;\theta)). \quad (4)$$

We optimize over $-\log D(I, f_T(I;\theta))$ instead of $\log(1 - D(I, f_T(I;\theta)))$ for better gradient performance [8].<br>
<br>
3. Exclusion loss<br>

$$L_{\text{excl}}(\theta) = \sum_{I\in\mathcal{D}} \sum_{n=1}^{N} \|\Psi(f_T^{\downarrow n}(I;\theta), f_R^{\downarrow n}(I;\theta))\|_F, \quad (5)$$

$$\Psi(T,R) = \tanh(\lambda_T|\nabla T|) \odot \tanh(\lambda_R|\nabla R|), \quad (6)$$

where $\lambda_T$ and $\lambda_R$ are normalization factors, $\|\cdot\|_F$ is the Frobenius norm, $\odot$ denotes element-wise multiplication, and $n$ is the image downsampling factor: the images $f_T$ and $f_R$ are downsampled by a factor of $2^{n-1}$ with bilinear interpolation. We set $N = 3$, $\lambda_T = \sqrt{\frac{\|\nabla R\|_F}{\|\nabla T\|_F}}$, and $\lambda_R = \sqrt{\frac{\|\nabla T\|_F}{\|\nabla R\|_F}}$ in our experiments.<br>
<br>
4. All<br>

$$L(\theta) = w_1 L_{\text{feat}}(\theta) + w_2 L_{\text{adv}}(\theta) + w_3 L_{\text{excl}}(\theta), \quad (1)$$

where we set $w_1 = 0.1$, $w_2 = 0.01$ and $w_3 = 1$ to balance the weight of each term.<br>

$$L_R(\theta) = \sum_{(I,R)\in\mathcal{D}} \|f_R(I;\theta) - R\|_1. \quad (7)$$

We train the network $f$ by minimizing $(L + L_R)$ on synthetic and real data jointly. Note that we disable $L_R$ when training on a real-world image as it is difficult to estimate $R$ precisely. We tried computing $R = I - T$ but $R$ sometimes contains significant artifacts because $I = R + T$ may not hold when $I$ is overexposed.<br>
<br>
三、Data<br>
<table>
<tr><th></th><th>Synthetic</th><th></th><th>Real</th><th></th></tr>
<tr><th>Method</th><th>SSIM</th><th>PSNR</th><th>SSIM</th><th>PSNR</th></tr>
<tr><td>Input</td><td>0.689</td><td>15.09</td><td>0.697</td><td>17.66</td></tr>
<tr><td>Pix2pix [12]</td><td>0.583</td><td>14.47</td><td>0.648</td><td>16.92</td></tr>
<tr><td>Li and Brown [21]</td><td>0.742</td><td>15.30</td><td>0.750</td><td>18.29</td></tr>
<tr><td>CEILNet [5]</td><td>0.826</td><td>20.47</td><td>0.762</td><td>19.04</td></tr>
<tr><td>Ours</td><td>0.853</td><td>22.63</td><td>0.821</td><td>21.30</td></tr>
</table>
Table 1: Quantitative comparison results among our method and 3 other previous methods. We evaluated on synthetic data provided by CEILNet [5], and our real image test set. We also provide a trivial baseline that takes the input image as the result transmission image.<br>
<br>
<table>
<tr><th></th><th>Synthetic</th><th></th><th>Real</th><th></th></tr>
<tr><th>Method</th><th>SSIM</th><th>PSNR</th><th>SSIM</th><th>PSNR</th></tr>
<tr><td>Ours w/o $L_{\text{feat}}$</td><td>0.683</td><td>18.24</td><td>0.743</td><td>19.07</td></tr>
<tr><td>Ours w/o $L_{\text{adv}}$</td><td>0.818</td><td>20.80</td><td>0.793</td><td>21.12</td></tr>
<tr><td>Ours w/o $L_{\text{excl}}$</td><td>0.796</td><td>19.58</td><td>0.802</td><td>20.22</td></tr>
<tr><td>Ours $L_{\text{adv}}$-only</td><td>0.765</td><td>18.05</td><td>0.782</td><td>19.52</td></tr>
<tr><td>Ours complete</td><td>0.853</td><td>22.63</td><td>0.821</td><td>21.30</td></tr>
</table>
Table 3: Quantitative comparisons on synthetic and real images among multiple ablated models of our method. We remove each of the three losses and evaluate on the re-trained models. 'Ours $L_{\text{adv}}$-only' denotes our method trained with only an adversarial loss. Our complete model shows better performance on both synthetic and real data. We evaluate on synthetic data provided by CEILNet [5], and our real test images described in Section 5.2.
</td>
<td>1. Feature loss makes use of the image feature information; Exclusion loss makes use of the edge information on pixel level; Use GAN to generate a loss function; And all of this three loss functions is inspiring and creative.<br><br>2. Use tanh as jump function to minimize the correlation between the predicted transmission and reflection layers in the gradient domain.</td>
<td>Learn the background of the research</td>
</tr>
</table>

CRRN:Multi-Scale Guided Concurrent Reflection Removal Network

**Article**

Wan, R., Shi, B., Duan, L. Y., Tan, A. H., & Kot, A. C. (2018). Crrn: Multi-scale guided concurrent reflection removal network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 4777-4785).

**Key**

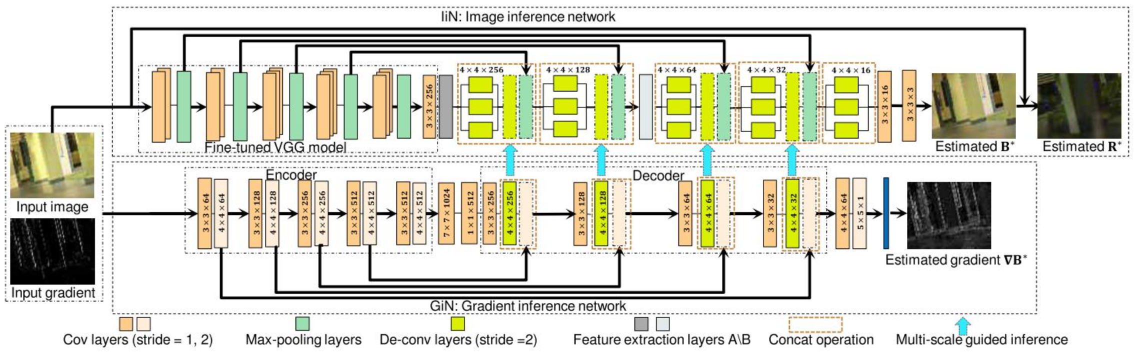
**Main:**
Propose a network integrating image appearance information and multi-scale gradient information with human perception inspired loss function

**Problems and Methods:**
single image reflection removal (layer separation):
A network makes use of both gradient features and image appearance features

**Background:**
The old methods are simply infeasible in many practical situations.

**Data**

一、Network Structure




GiN is designed to learn the gradient features, while IiN uses a well-trained VGG16 the 'Reduction- A/B layers' from Inception-ResNet-v2 to extract appearance features. Moreover, a residual network structure is applied at the end of the network to get the R.

二、**Training(synthesis dataset)**

GiN(independent, learning rate = 0.0001, epochs = 40) --> entire network(learining rate = 0.0001, epochs = 50) --> entire network(learning rate = 0.00001, epochs = 30)

**Loss for IiN:**

$$\mathbf{SSIM}(x, x^\star) = \frac{(2\mu_x \mu_{x^\star} + C_1)(2\sigma_{xx^\star} + C_2)}{(\mu_x^2 + \mu_{x^\star}^2 + C_1)(\sigma_x^2 + \sigma_{x^\star}^2 + C_2)}, \quad (3)$$

where $\mu_x$ and $\mu_x^\star$ are the means of $x$ and $x^\star$, $\sigma_x$ and $\sigma_{x^\star}$ are the variances of $x$ and $x^\star$, and $\sigma_{xx^\star}$ is their corresponding covariances. SSIM measures the similarity between two

$$\mathcal{L}^{\mathbf{SSIM}}(x, x^\star) = 1 - \mathbf{SSIM}(x, x^\star), \quad (4)$$

SSIM may cause changes of brightness and shifts of colors which makes the final results become dull. Therefore, MAE is used as L1 loss.

**Loss for GiN:**

In GiN, the luminance and contrast components in SSIM become undefined. We therefore omit the dependence of contrast and luminance in the original SSIM and define the loss function for GiN as

$$\mathcal{L}^{\mathbf{SI}}(x, x^\star) = 1 - \mathbf{SI}(x, x^\star). \quad (5)$$

SI is used to measure the structural similarity between two images as demonstrated in [27], which is defined as

$$\mathbf{SI} = \frac{2\sigma_{xx^\star} + c}{\sigma_x^2 + \sigma_{x^\star}^2 + c}, \quad (6)$$

**Loss for all:**

$$\begin{aligned}\mathcal{L} =& \gamma \mathcal{L}^{\mathbf{SSIM}}(\mathbf{B}, \mathbf{B}^\star) + \mathcal{L}_1(\mathbf{B}, \mathbf{B}^\star) + \\ & \mathcal{L}^{\mathbf{SSIM}}(\mathbf{R}, \mathbf{R}^\star) + \mathcal{L}^{\mathbf{SI}}(\nabla\mathbf{B}, \nabla\mathbf{B}^\star),\end{aligned} \quad (7)$$

where the weighting coefficient $\gamma$ is empirically set as 0.8 in our experiments.

三、Data

**Subscript 'r' represents the 'regional'**

| | SSIM | SI | $\text{SSIM}_r$ | $\text{SI}_r$ |
|---|---|---|---|---|
| Ours | **0.895** | **0.925** | **0.861** | **0.890** |
| FY17 [7] | 0.867 | 0.902 | 0.812 | 0.847 |
| NR17 [1] | 0.884 | 0.903 | 0.850 | 0.880 |
| WS16 [28] | 0.876 | 0.910 | 0.843 | 0.881 |
| LB14 [16] | 0.833 | 0.920 | 0.801 | 0.861 |

Table 2. Result comparisons of the proposed CRRN against CRRN using $\mathcal{L}_1$ loss in Equation (7) only and its sub-networks.

| | SSIM | SI | $\text{SSIM}_r$ | $\text{SI}_r$ |
|---|---|---|---|---|
| IiN in CRRN | **0.895** | **0.925** | **0.861** | **0.890** |
| IiN in CRRN ($\mathcal{L}_1$) | 0.883 | 0.910 | 0.849 | 0.865 |
| IiN only | 0.867 | 0.892 | 0.843 | 0.859 |

**Comments**

**TODO:**

1. Use SSIM as loss function, instead of evaluation, which may not be better than the perceptual loss in Single image reflection separation with perceptual losses in my opinion. This needs experiments.

2. Understanding why 'SSIM may cause changes of brightness and shifts of colors which makes the final results become dull.' --another paper needs to be read **'Loss functions for image restoration with neural networks'**

**Idea:**

1. The concatenation of CNN layers to enhance connection.

2. Estimate edge map, reflection image and background image with only a single network structure.

**Why**

Learn the background of the research

BDN:reflection separation with a deep and bidirectional network

**Article**

Yang, J., Gong, D., Liu, L., & Shi, Q. (2018). Seeing deeply and bidirectionally: A deep learning approach for single image reflection removal. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 654-669).

**Key**

**Main:**
Propose a cascade deep neural network, which estimates both the background image and the reflection and makes use of the result to refine the estimation.

It considers the case when the reflections do not have strong blurry or have similar brightness and structure with the background.

**Problems and Methods:**
single image reflection removal (layer separation):
A cascade deep neural network based on the assumption that reflection may not be blurred.

**Background:**
The old methods are simply infeasible in many practical situations.

**Data**

**一、Network Structure**

14 layers(G0) --> 10 layers(H) --> 10 layers(G1)

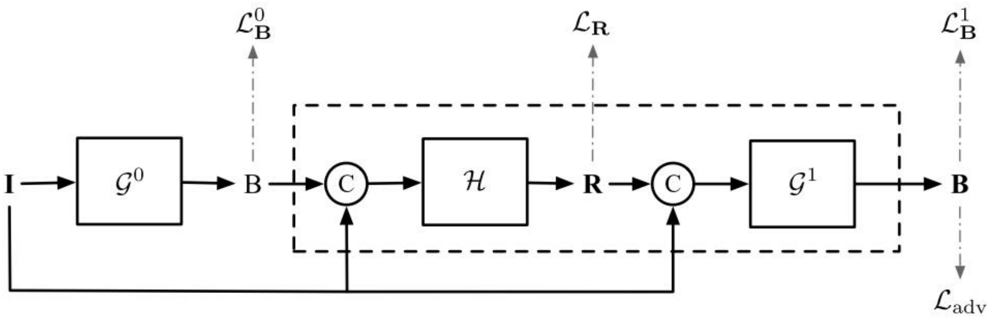



**Fig. 2.** Overview of our proposed BDN network architecture and the training objectives. Component C stands for tensor concatenation.

Deep Bidirectional Estimation for Single Image Reflection Removal 5

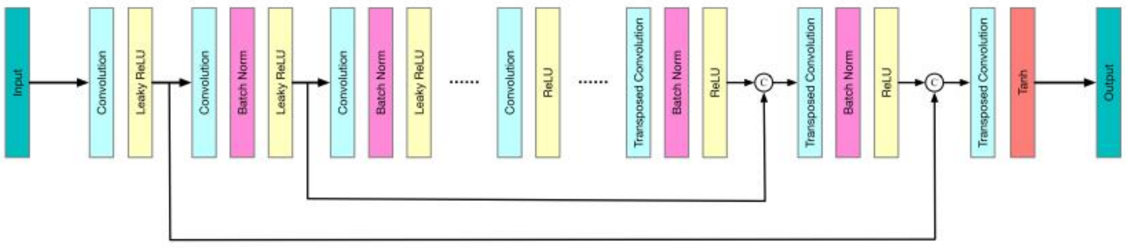



**Fig. 3.** The network structure of $\mathcal{G}^0$, $\mathcal{H}$ and $\mathcal{G}^1$. C stands for tensor concatenation.

**Encoder:** all convolution layers are followed by BatchNorm layer and leaky ReLU with slope 0.2, except for the first convolution layer

**Decoder:** each transposed convolution with stride 2 which upsamples the feature

- All the kernel size of the filters are 4*4 and the skip connections concatenate each channel from layer i to layer n-i where n is the number of layers

**二、Trainning(synthesis dataset)**

Training end to end and training independently are both tested.

**Loss for pixel-wise difference:**

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

where

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

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

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

**Loss for perceptual difference(by GAN):**

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

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

**Loss for all:**

**Full objective** Finally, we sum the $\ell_2$ loss and adversarial loss as the final objective:

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

where $\lambda$ is the hyper-parameter that controls the relative importance of the two objectives.

**三、Data**

**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 |

**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 |

**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** |

**Comments**

1. Bidirectional network.

2. Adversarial loss like ‘Single image reflection separation with perceptual losses’.

3. Consider the case that reflections may not have strong blurry.

**Why**

Learn the background of the research

ERRN:reflection removal exploiting misaligned training data and network enhancements

**Article**

Wei, K., Yang, J., Fu, Y., Wipf, D., & Huang, H. (2019). Single image reflection removal exploiting misaligned training data and network enhancements. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 8178-8187).

**Key**

**Main:**

Augment a baseline network architecture by embedding context encoding modules and introduce an alignment-invariant loss function that facilitates exploiting misaligned real-world training data

**Problems and Methods:**

single image reflection removal (layer separation):

Propose to leverage a network architecture that is sensitive to contextual information to better tackle the intrinsic ill-posedness and diminish ambiguity.

Seek to expand the sources of viable training data by facilitating the use of misaligned training pairs by a novel loss function, which are considerably easier to collect.

**Background:**

The old methods are simply infeasible in many practical situations.

**Data**

**一、Network Structure**

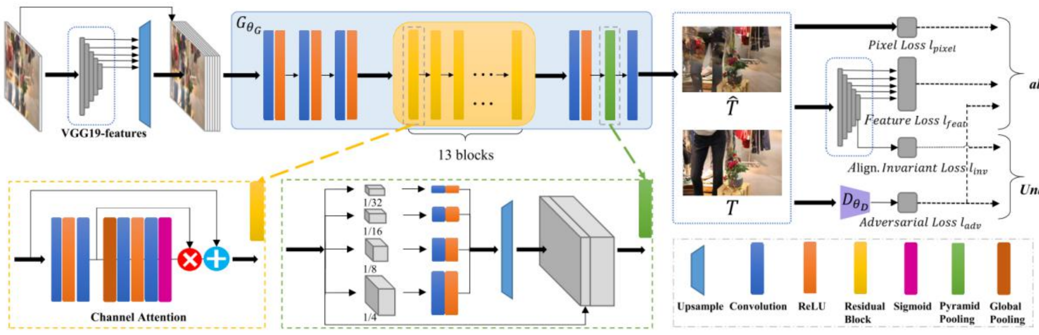



Figure 2: Overview of our approach for single image reflection removal.

**Channel-wise context(the yellow block):**

convolution + ReLU --> convolution(C*(H*W) feature maps) --> global pooling(average pooling, C*1) --> convolution(downsample, R*1 R<C) + ReLU --> convolution(upsample, C*1) + Sigmoid --> calibrate feature maps as channel-special gates

**Multi-scale spatial context(the green block):**

Pooling at 4, 8, 16, 32 --> convolution + ReLU(each scale separately) --> upsampled via bilinear interpolation --> concatenate

**Tips:**

- BN sometimes can lead to considerably worse performance, including color attenuation/shifting issues as sometimes observed in image-to-image translation tasks.

**二、Trainning(synthesis dataset)**

**Loss for aligned data(reference):**

**Pixel loss.** Following [5], we penalize the pixel-wise intensity difference of $T$ and $\hat{T}$ via $l_{pixel} = \alpha\|\hat{T} - T\|_2^2 + \beta(\|\nabla_x\hat{T} - \nabla_x T\|_1 + \|\nabla_y\hat{T} - \nabla_y T\|_1)$ where $\nabla_x$ and $\nabla_y$ are the gradient operator along x- and y-direction, respectively. We set $\alpha = 0.2$ and $\beta = 0.4$ in all our experiments.

**Feature loss.** We define the feature loss based on the activations of the 19-layer VGG network [33] pretrained on ImageNet [29]. Let $\phi_l$ be the feature from the $l$-th layer of VGG-19, we define the feature loss as $l_{feat} = \sum_l \lambda_l\|\phi_l(T) - \phi_l(\hat{T})\|_1$ where $\{\lambda_l\}$ are the balancing weights. Similar to [47], we use the layers 'conv2_2', 'conv3_2', 'conv4_2', and 'conv5_2' of VGG-19 net.

**Adversarial loss.** We further add an adversarial loss to improve the realism of the produced background images. We define an opponent discriminator network $D_{\theta_D}$ and minimize the relativistic adversarial loss [18] defined as $l_{adv} = l_{adv}^G = -\log(D_{\theta_D}(T,\hat{T})) - \log(1 - D_{\theta_D}(\hat{T},T))$ for $G_{\theta_G}$ and $l_{adv}^D = -\log(1 - D_{\theta_D}(T,\hat{T})) - \log(D_{\theta_D}(\hat{T},T))$ for $D_{\theta_D}$ where $D_{\theta_D}(T,\hat{T}) = \sigma(C(T) - C(\hat{T}))$ with $\sigma(\cdot)$ being the sigmoid function and $C(\cdot)$ the non-transformed discriminator function (refer to [18] for details).

To summarize, our loss for aligned data is defined as:

$$l_{aligned} = \omega_1 l_{pixel} + \omega_2 l_{feat} + \omega_3 l_{adv} \quad (2)$$

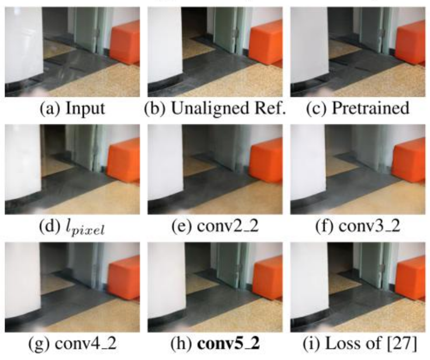
where we empirically set the weights as $\omega_1 = 1, \omega_2 = 0.1$, and $\omega_3 = 0.01$ respectively throughout our experiments.

**Loss for misaligned data:**

**Alignment-invariant loss.** Based on the above study, we now formally define our invariant loss component designed for unaligned data as $l_{inv} = \|\phi_h(T) - \phi_h(\hat{T})\|_1$, where $\phi_h$ denotes the 'conv5_2' feature of the pretrained VGG-19 network. For unaligned data, we also apply an adversarial loss which is not affected by misalignment. Therefore, our overall loss for unaligned data can be written as

$$l_{unaligned} = \omega_4 l_{inv} + \omega_5 l_{adv} \quad (3)$$

where we set the weights as $\omega_4 = 0.1$ and $\omega_5 = 0.01$.



(a) Input (b) Unaligned Ref. (c) Pretrained (d) $l_{pixel}$ (e) conv2_2 (f) conv3_2 (g) conv4_2 (h) **conv5_2** (i) Loss of [27]

Figure 3: The effect of using different loss to handle misaligned real data. (a) and (b) are the unaligned image pair $(I,T)$. (c) shows the reflection removal result of our network trained on synthetic data and a small number of aligned real data (see Section 4 for details). Reflection can still be observed in the predicted background image. (d) is the result finetuned on $(I,T)$ with pixel-wise intensity loss. (e)-(h) are the results finetuned with features at different layers of VGG-19. Only the highest-level feature from 'conv5_2' yields satisfactory result. (i) shows the results finetuned with the loss of [27]. (**Best viewed on screen with zoom**)

**三、Data**

| | Synthetic | | Real20 | |
|---|---|---|---|---|
| Model | PSNR | SSIM | PSNR | SSIM |
| CEILNet-F [5] | 24.70 | 0.884 | 20.32 | 0.739 |
| BaseNet only | 25.71 | 0.926 | 21.51 | 0.780 |
| BaseNet + CSC | 27.64 | 0.940 | 22.61 | 0.796 |
| BaseNet + MSC | 26.03 | 0.928 | 21.75 | 0.783 |
| ERRNet | **27.88** | **0.941** | **22.89** | **0.803** |

| Training Scheme | PSNR | SSIM |
|---|---|---|
| Synthetic only | 19.79 | 0.741 |
| + 50 aligned | 22.00 | 0.785 |
| + 90 aligned | 22.89 | 0.803 |
| + 50 aligned, + 40 unaligned trained with: | | |
| $l_{pixel}$ | 21.85 | 0.766 |
| $l_{inv}$ | 22.38 | **0.797** |
| $l_{cx}$ | **22.47** | 0.796 |
| $l_{inv} + l_{cx}$ | 22.43 | 0.796 |

Table 3: Quantitative results of different methods on four real-world benchmark datasets. The best results are indicated by red color and the second best results are denoted by blue color. The results of 'Average' are obtained by averaging the metric scores of all images from these four real-world datasets.

| Dataset | Index | Methods: Input | LB14 [25] | CEILNet [5] | CEILNet F | Zhang *et al.* [47] | BDN [44] | BDN F | ERRNet |
|---|---|---|---|---|---|---|---|---|---|
| Real20 | PSNR | 19.05 | 18.29 | 18.45 | 20.32 | 21.89 | 18.41 | 20.06 | 22.89 |
| | SSIM | 0.733 | 0.683 | 0.690 | 0.739 | 0.787 | 0.726 | 0.738 | 0.803 |
| | NCC | 0.812 | 0.789 | 0.813 | 0.834 | 0.903 | 0.792 | 0.825 | 0.877 |
| | LMSE | 0.027 | 0.033 | 0.031 | 0.028 | 0.022 | 0.032 | 0.027 | 0.022 |
| Objects | PSNR | 23.74 | 19.39 | 23.62 | 23.36 | 22.72 | 22.73 | 24.00 | 24.87 |
| | SSIM | 0.878 | 0.786 | 0.867 | 0.873 | 0.879 | 0.856 | 0.893 | 0.896 |
| | NCC | 0.981 | 0.971 | 0.972 | 0.974 | 0.964 | 0.978 | 0.978 | 0.982 |
| | LMSE | 0.004 | 0.007 | 0.005 | 0.005 | 0.005 | 0.005 | 0.004 | 0.003 |
| Postcard | PSNR | 21.30 | 14.88 | 21.24 | 19.17 | 16.85 | 20.71 | 22.19 | 22.04 |
| | SSIM | 0.878 | 0.795 | 0.834 | 0.793 | 0.799 | 0.859 | 0.881 | 0.876 |
| | NCC | 0.947 | 0.929 | 0.945 | 0.926 | 0.886 | 0.943 | 0.941 | 0.946 |
| | LMSE | 0.005 | 0.008 | 0.008 | 0.013 | 0.007 | 0.005 | 0.004 | 0.004 |
| Wild | PSNR | 26.24 | 19.05 | 22.36 | 22.05 | 21.56 | 22.36 | 22.74 | 24.25 |
| | SSIM | 0.897 | 0.755 | 0.821 | 0.844 | 0.836 | 0.830 | 0.872 | 0.853 |
| | NCC | 0.941 | 0.894 | 0.918 | 0.924 | 0.919 | 0.932 | 0.922 | 0.917 |
| | LMSE | 0.005 | 0.027 | 0.013 | 0.009 | 0.010 | 0.009 | 0.008 | 0.011 |
| *Average* | PSNR | 22.85 | 17.51 | 22.30 | 21.41 | 20.22 | 21.70 | 22.96 | 23.59 |
| | SSIM | 0.874 | 0.781 | 0.841 | 0.832 | 0.838 | 0.848 | 0.879 | 0.879 |
| | NCC | 0.955 | 0.937 | 0.948 | 0.943 | 0.925 | 0.951 | 0.950 | 0.956 |
| | LMSE | 0.006 | 0.011 | 0.009 | 0.010 | 0.007 | 0.007 | 0.006 | 0.005 |

**Comments**

1. Using pyramid scale which has proven to be an effective global-scene-level representation in semantic segmentation, according to **'Pyramid scene parsing network'**.

2. Intuitively, the deeper the feature, the more likely it is to be insensitive to misalignment, which inspire them use con5_2 from VGG19 as the misaligned loss.

3. The experiments are in detail.

**Why**

Learn the background of the research

Reflection removal with DP sensors

**Article**

Punnappuranth, A., & Brown, M. S. (2019). Reflection removal using a dual-pixel sensor. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 1556-1565).

**Key**

**Main:**
Introduce a new reflection removal method that exploits the two sub-aperture views available on a DP sensor.

**Problems and Methods:**
single image reflection removal (layer separation):
Use the information available from dual pixel (DP) sensors to remove reflection interface by constructing a model of DP sensors and solve the optimization problem.

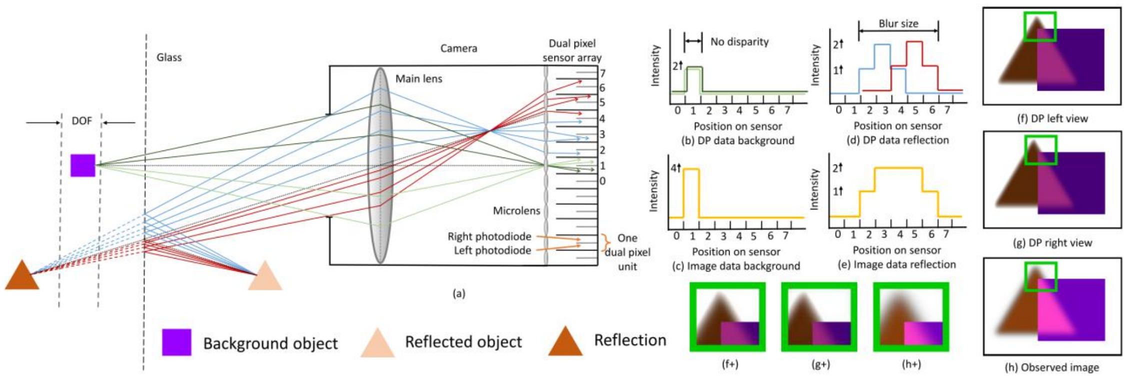
**Background:**
The old methods are simply infeasible in many practical situations.

**Data**

**一、DP Model**

**Two assumptions:**

- the background layer has predominately stronger image intensity than the reflection layer
- the background scene content lies within the depth of field (DOF) of the camera, while the objects in the scene being reflected on the glass are at a different depth and therefore outside the DOF




$$\mathbf{g}_{\text{LV}} = \frac{\mathbf{b}}{2} + \mathbf{W}_{\text{LV}}\mathbf{f}, \ \mathbf{g}_{\text{RV}} = \frac{\mathbf{b}}{2} + \mathbf{W}_{\text{RV}}\mathbf{f}, \qquad (1)$$

where $\mathbf{W}_{\text{LV}}$ and $\mathbf{W}_{\text{RV}}$ are the matrices that multiply the underlying sharp reflection layer $\mathbf{f}$ to produce its defocused and shifted versions of half intensity in the left and right views, respectively. The observed image $\mathbf{g}$ can be expressed as $\mathbf{g} = \mathbf{g}_{\text{LV}} + \mathbf{g}_{\text{RV}} = \mathbf{b} + \mathbf{r}$, where $\mathbf{r}$ equals the blurred reflection layer and is given by $\mathbf{r} = (\mathbf{W}_{\text{LV}} + \mathbf{W}_{\text{RV}})\mathbf{f}$.

**二、Algorithm**

1. Get two gradient maps through derivative filters hl and hr
2. Select a N*N patch from hl and perform a horizontal search over a range of −t to t pixels in hr

**3. Get the background gradient map**

Following [30], we find the minimum of these $2t+1$ points and fit a quadratic $\frac{1}{2}a_1x^2+a_2x+a_3$ to the SSD value using the minimum and its two surrounding points. At a given pixel $i$, the location of the quadratic's minimum $s_i = \frac{-a_2}{a_1}$ serves as our sub-pixel minimum. We also compute a confidence value at each pixel $i$ as [4]:

$$\beta_i = \exp\left(\frac{\log|a_{1_i}|}{\sigma_{a_1}} - \frac{a_{3_i}}{\sigma_{a_3}^2}\right). \qquad (2)$$

We construct our weighted gradient map of the background using the confidence values $\beta_i$ as

$$c_i = \begin{cases} \rho\beta_i & \text{if } |s_i| < \epsilon \text{ and } \beta_i > 1, \\ 0 & \text{otherwise.} \end{cases} \qquad (3)$$

Two examples of our estimated background gradient maps are shown in Fig. 3. We fix $\rho = 5$, $N = 11$, $t = 5$, $\sigma_{a_1} = 5$, and $\sigma_{a_3} = 256$ for all examples presented in this paper.

Following [22], we model the blurred reflection layer's gradient distribution using a Gaussian function with a narrow spread as

$$P_R(l) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{l^2}{2\sigma^2}}, \qquad (4)$$

where $l$ represents the gradient value, and $\sigma$ denotes the standard deviation of the Gaussian.

It is well known that the gradients of natural images have a heavy-tailed distribution, and that this distribution can be modeled using a hyper-Laplacian function [16, 17]. Therefore, we express the probability distribution of the gradients of the in-focus background layer as

$$P_B(l) = e^{-\alpha|l|^p}, \qquad (5)$$

where $\alpha$ is a positive scalar, and we set $p = \frac{2}{3}$ [16].

**4. Cost Function**

Specifically, we maximize the joint probability $P(\mathbf{b}, \mathbf{r})$. Assuming that the background and the reflection are independent [19], the joint probability can be expressed as the product of the probabilities of each of the two layers – that is, $P(\mathbf{b}, \mathbf{r}) = P(\mathbf{b})P(\mathbf{r})$. Following [36], we define our distribution over both background and reflection layers using the histogram of derivative filters as

$$P(\mathbf{z}) \approx \prod_{i,k} P\big((\mathbf{D}_k\mathbf{z})_i\big), \quad \mathbf{z} = \text{ either } \mathbf{b} \text{ or } \mathbf{r}, \qquad (6)$$

where we assume that the horizontal and vertical derivative filters $\mathbf{D}_k \in \{\mathbf{D}_x, \mathbf{D}_y, \mathbf{D}_{xx}, \mathbf{D}_{xy}, \mathbf{D}_{yy}\}$ are independent over space and orientation.

Maximizing $P(\mathbf{b}, \mathbf{r})$ is equal to minimizing its negative *log*, and from equations (4)(5)(6), we obtain the following cost function:

$$\arg\min_{\mathbf{b},\mathbf{r}} \left\{ \sum_{i,k} \left( |(\mathbf{D}_k\mathbf{b})_i|^p + \lambda\big((\mathbf{D}_k\mathbf{r})_i\big)^2 \right) \right\}, \qquad (7)$$

where we integrate the relative weight between the two terms and the multiplier $\frac{1}{2\alpha\sigma^2}$ into a single parameter $\lambda$, which controls the amount of defocus blur in the reflection layer. This can be rewritten as

$$\arg\min_{\mathbf{b},\mathbf{r}} \left\{ ||\mathbf{D}\mathbf{b}||_p^p + \lambda||\mathbf{D}\mathbf{r}||_2^2 \right\}, \qquad (8)$$

where the matrix $\mathbf{D}$ consists of the five $\mathbf{D}_k$s vertically stacked. Expressing in terms of a single layer $\mathbf{b}$, and incorporating the confidence values $\mathbf{C} = \text{diag}(c_i)$ from equation (3) to enforce agreement with the labeled gradients, we obtain

$$\arg\min_{\mathbf{b}} \left\{ ||\mathbf{D}\mathbf{b}||_p^p + \lambda||\mathbf{C}\mathbf{D}(\mathbf{g} - \mathbf{b})||_2^2 \right\}. \qquad (9)$$

**三、Data**

**Algorithm 1** Reflection removal using a dual-pixel sensor

**Input:** : Input image $\mathbf{g}$, the left $\mathbf{g}_{\text{LV}}$ and right $\mathbf{g}_{\text{RV}}$ DP views, relative weight $\lambda$, maximum iterations $Q$.
**Output:** : Background $\mathbf{b}$, and blurred reflection $\mathbf{r}$.

```
Compute C using g_LV and g_RV (see Section 3.1)
D_* = CD
q = 0
b = (D^T D + λD_*^T D_*)^(-1) (λD_*^T D_* g)
do
    e_i = (max(|(Db)_i|, 0.001))^(p-2)
    E = diag(e_i)
    b = (D^T E D + λD_*^T D_*)^(-1) (λD_*^T D_* g)
    q++
while q < Q
r = g − b
```

| Method | PSNR (dB) | SSIM | sLMSE | NCC | SI |
|---|---|---|---|---|---|
| LB13 [21] | 16.12 | 0.689 | 0.870 | 0.966 | 0.758 |
| GC14 [11] | 16.02 | 0.798 | 0.888 | 0.945 | 0.496 |
| LB14 [22] | 14.20 | 0.842 | 0.797 | 0.981 | 0.840 |
| WS16 [33] | 16.62 | 0.836 | 0.884 | 0.975 | 0.837 |
| ZN18 [40] | 15.57 | 0.797 | 0.867 | 0.979 | 0.818 |
| YG18 [39] | 16.49 | 0.832 | 0.871 | 0.978 | 0.847 |
| Ours | **19.45** | **0.883** | **0.946** | **0.982** | **0.870** |

**Comments**

1. This kind of method can not handle the reflection removal problem with strong reflection either.

2. A heuristic method to compute confidence.

3. The assumption that the reflection layer is out of focus is not appropriate in every situation.

4. The new idea with a special sensor to solve the problem is inspiring.

**Why**

Learn the background of the research

IBCLN:reflection removal with LSTM network

**Article**

Li, C., Yang, Y., He, K., Lin, S., & Hopcroft, J. E. (2020). Single image reflection removal through cascaded refinement. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 3565-3574).

**Key**

**Main:**

Propose an Iterative Boost Convolutional LSTM Network (IBCLN) that enables cascaded prediction for reflection removal.

**Problems and Methods:**

single image reflection removal (layer separation):

Use a cascaded network to iterate the computation to make use of the previous result.

Construct a convolutional LTSM network which saves information from the previous iteration and and allows gradients to flow unchanged to do with the gradient vanishment and limited training guidance at each step.

**Background:**

The old methods are simply infeasible in many practical situations.

**Data**

**一、Network Structure**

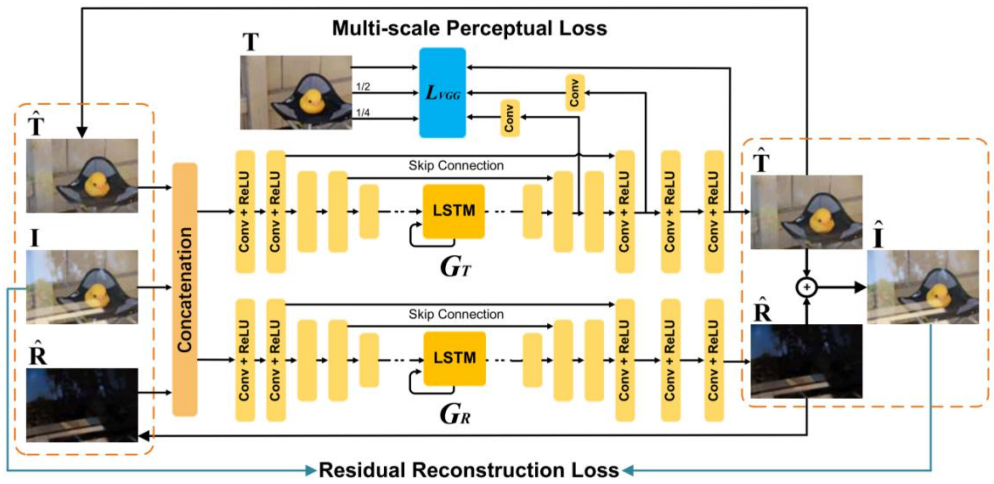



Figure 2. The architecture of IBCLN. The cascaded network consists of a transmission generative sub-network $G_T$ and a reflection generative sub-network $G_R$ with skip connections, both of which are convolutional LSTM networks. The images generated at each time step by the two sub-networks will be fed back at the next time step. The overall network is trained in an end-to-end manner.

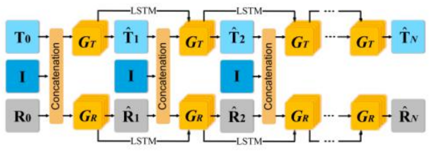



Figure 3. Characterizing IBCLN with increasing number of time steps. All blocks labeled as $G_T$ indicate one sub-network and all blocks labeled as $G_R$ indicate another sub-network. The output at time step $t-1$ serves as the input at time step $t$. $\hat{\mathbf{T}}_1, \hat{\mathbf{T}}_2, ..., \hat{\mathbf{T}}_N$ are the predicted transmission. $\hat{\mathbf{R}}_1, \hat{\mathbf{R}}_2, ..., \hat{\mathbf{R}}_N$ are the predicted residual reflection.

Sub-Network:

encoder(11 convolution layers each with ReLU) --> convolutional LSTM --> decoder(8 convolution layers each with ReLU)

**二、Trainning(synthetic dataset 2800 and real world dataset 1200)**

**Loss of residual reconstruction:**

$$\widetilde{\mathbf{R}} = \mathbf{I} - \alpha \cdot \mathbf{T}. \quad (1)$$

With this definition of $\widetilde{\mathbf{R}}$, the clipping operation is not needed and we avoid its loss of information. After $\widetilde{\mathbf{R}}$ is calculated, it can be used as the ground truth of $G_R$ to guide the generation of the predicted residual reflection $\hat{\mathbf{R}}$. Then, we can simply revert Eq. (1) in the objective function, as

$$\hat{\mathbf{I}} = \alpha \cdot \hat{\mathbf{T}} + \hat{\mathbf{R}}, \quad (2)$$

The residual reconstruction loss is defined as:

$$\mathcal{L}_{residual} = \sum_{I \in \mathcal{D}} \sum_{t=1}^{N} \mathcal{L}_{MSE}(\mathbf{I}, \hat{\mathbf{I}}_t). \quad (3)$$

$\mathcal{L}_{MSE}$ indicates the mean squared error. $t$ denotes the time step of the two sub-networks. $N$ represents the final time step when $\hat{\mathbf{T}}$ converges.

**Loss of multi-scale perceptual:**

$$\mathcal{L}_{MP} = \sum_{T, T^3, T^5 \in \mathcal{D}} (\mathcal{L}_{VGG}(\mathbf{T}, \hat{\mathbf{T}}) + \gamma_3 \mathcal{L}_{VGG}(\mathbf{T^3}, \hat{\mathbf{T}}^\mathbf{3}) + \gamma_5 \mathcal{L}_{VGG}(\mathbf{T^5}, \hat{\mathbf{T}}^\mathbf{5})), \quad (4)$$

where $\hat{\mathbf{T}}$, $\hat{\mathbf{T}}^\mathbf{3}$, $\hat{\mathbf{T}}^\mathbf{5}$ indicate the outputs of the last, $3^{rd}$ last and $5^{th}$ last layers at time step $N$, whose sizes are 1, $\frac{1}{2}$ and $\frac{1}{4}$ of the original size, respectively. $\mathbf{T}$, $\mathbf{T^3}$ and $\mathbf{T^5}$ indicate the ground truth that has the same scale as that of the outputs, respectively. Layers with smaller size are not considered since their information is relatively insignificant. We set $\gamma_3 = 0.8$ and $\gamma_5 = 0.6$. All the images are fed into the VGG19 network [21]. We compare the outputs of the layers 'conv1_2' and 'conv2_2' in the VGG19 network.

**Loss of pixel:**

$$\mathcal{L}_{pixel} = \sum_{T \in \mathcal{D}} \sum_{t=1}^{N} [\mathcal{L}_{MSE}(\mathbf{T}, \hat{\mathbf{T}}_t) + \mathcal{L}_{MSE}(\widetilde{\mathbf{R}}, \hat{\mathbf{R}}_t)], \quad (5)$$

where $\widetilde{\mathbf{R}}$ is the residual reflection. $\hat{\mathbf{T}}_t$ and $\hat{\mathbf{R}}_t$ are the outputs at time step $t$.

**Adversarial Loss:**

$$\mathcal{L}_{adv} = \sum_{T \in \mathcal{D}} -\log \boldsymbol{D}(\mathbf{T}, \hat{\mathbf{T}}). \quad (6)$$

**All:**

$$L = \lambda_1 \mathcal{L}_{residual} + \lambda_2 \mathcal{L}_{MP} + \lambda_3 \mathcal{L}_{pixel} + \lambda_4 \mathcal{L}_{adv}, \quad (7)$$

where we empirically set the weights as $\lambda_1 = 2, \lambda_2 = 1, \lambda_3 = 2, \lambda_4 = 0.01$ throughout our experiments.

**三、Data**

Table 1. Quantitative comparison of different methods on three real-world benchmark datasets. The best results are in **bold** and orange color, and the second best results are underlined and in blue color. 'Average' is obtained by averaging the metric scores of all images from all the above real-world datasets.

| Dataset (size) | Index | CEILNet-F [4] | Zhang et al. [34] | BDN-F [33] | RmNet [31] | ERRNet-F [30] | IBCLN |
|---|---|---|---|---|---|---|---|
| Object (200) | PSNR | 22.81 | 22.68 | 23.02 | 20.33 | 24.85 | **24.87** |
| | SSIM | 0.801 | 0.874 | 0.853 | 0.793 | 0.889 | **0.893** |
| Postcard (199) | PSNR | 20.08 | 16.81 | 20.71 | 19.71 | 21.99 | **23.39** |
| | SSIM | 0.810 | 0.797 | 0.857 | 0.808 | 0.874 | **0.875** |
| Wild (55) | PSNR | 22.14 | 21.52 | 22.34 | 21.98 | 24.16 | **24.71** |
| | SSIM | 0.819 | 0.829 | 0.821 | 0.821 | 0.847 | **0.886** |
| Zhang et al. (20) | PSNR | 18.79 | 22.42 | 19.47 | 18.77 | **23.35** | 21.86 |
| | SSIM | 0.749 | 0.792 | 0.720 | 0.681 | **0.811** | 0.762 |
| Nature (20) | PSNR | 19.33 | 19.56 | 18.92 | 19.36 | 22.18 | **23.57** |
| | SSIM | 0.745 | 0.736 | 0.737 | 0.725 | 0.756 | **0.783** |
| *Average* (494) | PSNR | 21.31 | 20.85 | 21.68 | 20.19 | 23.45 | **24.08** |
| | SSIM | 0.806 | 0.829 | 0.841 | 0.795 | 0.870 | **0.875** |

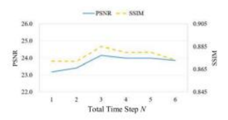



Figure 8. Results using different total time steps $N$ in IBCLN on $SIR^2$ [26]. Total time steps $N = 3$ yields the best performance.

Table 2. Ablation study of IBCLN for architecture on three testing sets. w/o $G_R$ means training with only one sub-network $G_T$. w/o iteration means the total time steps is 1. Each term contributes to the SIRR performance, and combining all achieves the best results.

| Model | Nature PSNR | Nature SSIM | Zhang et al. PSNR | Zhang et al. SSIM | $SIR^2$ PSNR | $SIR^2$ SSIM |
|---|---|---|---|---|---|---|
| w/o $G_R$ | 21.79 | 0.759 | 20.65 | 0.742 | 22.36 | 0.868 |
| w/o iteration | 21.82 | 0.764 | 20.49 | 0.739 | 23.09 | 0.872 |
| Complete | **23.57** | **0.783** | **21.86** | **0.762** | **24.20** | **0.884** |

Table 3. Ablation study of IBCLN for loss terms on three testing sets. Each loss contributes to IBCLN's performance, and combining all achieves the best result.

| Model | Nature PSNR | Nature SSIM | Zhang et al. PSNR | Zhang et al. SSIM | $SIR^2$ PSNR | $SIR^2$ SSIM |
|---|---|---|---|---|---|---|
| $L_{pixel}$ only | 21.98 | 0.739 | 19.54 | 0.722 | 22.91 | 0.843 |
| w/o $L_{adv}$ | 23.24 | 0.746 | 21.74 | 0.755 | 23.86 | **0.885** |
| w/o $L_{residual}$ | 22.54 | 0.770 | 20.98 | 0.755 | 23.74 | 0.881 |
| w/o $L_{MP}$ | 23.14 | 0.744 | 21.47 | 0.734 | 22.96 | 0.863 |
| Complete | **23.57** | **0.783** | **21.86** | **0.762** | **24.20** | 0.884 |

**Comments**

**TODO:**

1. Understand the skip connection according to **'Attentive generative adversarial network for rain- drop removal from a single image'**

2. Learn more about convolutional LSTM according to **'Convolutional lstm network: A machine learning approach for precipitation nowcasting'**

**Idea:**

1. Improve the reflection synthesis model.

2. Novel loss function residual reconstruction function.

3. Special network structure to iterate to get better result.

**Why**

Learn the background of the research