Chapter 8
Single Image Reflection Removal Using Deep Learning

Sushil Kumar, Peeyush Joshi, Vanita Garg, and Hira Zaheer

1 Introduction

It is often the case that the subject that we are trying to photograph is on the other side of the glass and we end up taking a photograph through a glass, as the glass in between is simply unavoidable or the hassle is not worth the efforts. Photographs, thus, taken contain undesirable reflections and degrade the visibility of the scene by blurring, obstructing or deforming the background scene and may result in failure or degradation of processing and analysing capabilities of computer-vision algorithms, such as object detection, event detection, object recognition, image segmentation, video tracking, etc. The problem of getting reflection-free images taken through glass is of great interest in the image processing and computer vision community and has practical demands.

$$\boldsymbol{I} = \boldsymbol{R} + \boldsymbol{B} \tag{8.1}$$

where
$\boldsymbol{I}$: n × m × 3 matrix which represents the reflection-contaminated image
$\boldsymbol{R}$: n × m × 3 matrix which represents the reflection layer
$\boldsymbol{B}$: n × m × 3 matrix which represents the background layer

S. Kumar (✉) · P. Joshi
Department of Computer Science & Engineering, National Institute of Technology Warangal, Hanamkonda, Telangana, India
e-mail: kumar.sushil@nitw.ac.in; pjoshi@student.nitw.ac.in

V. Garg · H. Zaheer
School of Basic and Applied Sciences, Galgotias University, Greater Noida, Uttar Pradesh, India

D. Singh et al. (eds.), *Design and Applications of Nature Inspired Optimization*, Women in Engineering and Science, https://doi.org/10.1007/978-3-031-17929-7_8

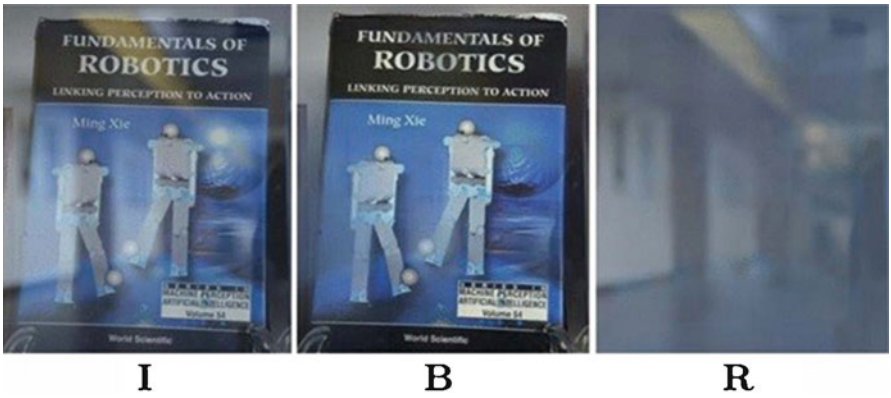

Fig. 8.1 Reflection-contaminated image "I", background layer "B" and reflection layer "R"

The goal of the work is to approximate the background layer $\boldsymbol{B}$ from the acquired image $\boldsymbol{I}$. Figure 8.1 illustrates the reflection-contaminated image as well as the ground truth for the background and the reflection layers.

The problem of removing reflection from a single image is ill-posed as, for a given reflection-contaminated image, there could be infinite possible decompositions into the background layer and the reflection layer; the same is illustrated with the help of an example image in Fig. 8.2. Also, lack of sufficient labelled data for training and reflection and background layers containing data from natural scenes adds to the ill-posedness of the problem.

Most of the existing methods to remove reflection use specialized hardware or multiple images to make the problem less ill-posed and produce. Recently, some research works used deep learning methods, which outperform the existing methods, but, still, they use very complex architectures and blur out or degrade the quality of the images and fail in cases when the background and the reflection layers are very similar in terms of brightness and structural appearance.

Our contributions to address the above-mentioned issues are as follows:

- We have trained a relatively simpler architecture end-to-end neural network to estimate the background layer.
- We have created a loss function based on SSIM score, which is better suited when comparing the similarities between images.
- We have created a larger labelled training dataset using data from multiple sources.

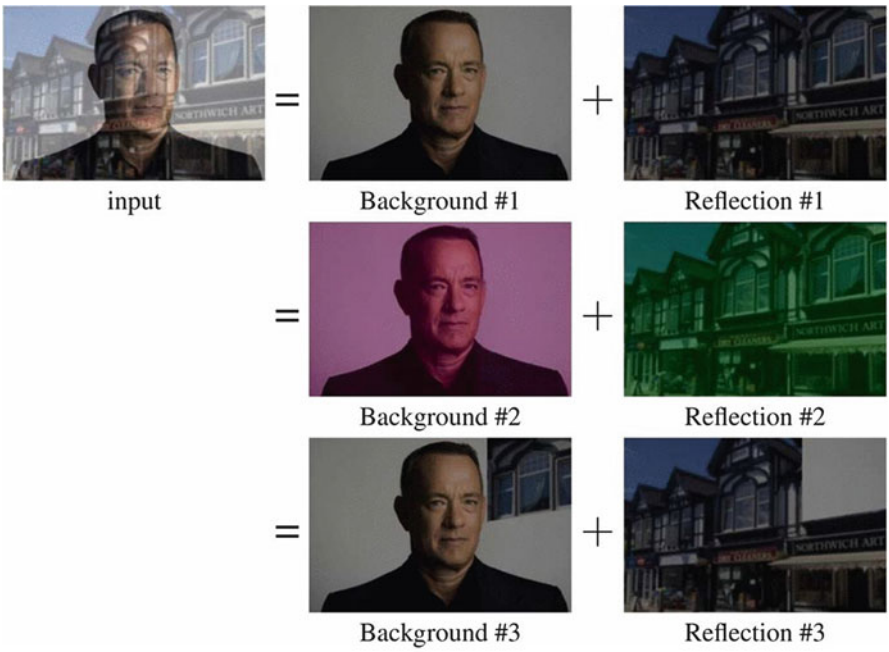

Fig. 8.2 Three possible separations of a reflection-contaminated image into the background and the reflection layers

2 Literature Survey

The problem of how to remove reflection artefacts from an image has been widely researched in the image processing and computer vision community. Existing work can be classified into two categories based on the number of inputs required to produce a single reflection-free image. The first category includes methods requiring multiple inputs (such as multiple images or the use of specialized hardware to capture the image), and the second category includes methods requiring single image as the input. Single image methods can be further classified based on the approach they use to solve the problem, conventional mathematical approaches or learning-based approaches.

Multi-image Methods

Multiple related images can be used to make the problem of reflection removal less ill-posed and easier to solve but make the process of capturing images difficult. Guo et al. [1] and Y. Li and M. S. Brown [2] use images taken from slightly different angles or video sequence. Agrawal et al. [3] use image pairs taken with and without

firing flash. Schechner et al. [4] use polarizer to obtain multiple polarized images. Kong et al. [5] use image pairs with the subject in and out of focus. These methods produce state-of-the-art results but are highly limited in practicability due to the complex process of capturing the images.

Single Input Methods

When compared with multi-image approaches, trying to suppress or remove reflection artefacts from a single image is difficult because of the constrained data.

Traditional Approaches

The following approaches use conventional mathematical approaches to remove the reflection.

Levin et al. [6] proposed an oversimplified approach based on local features of corners and edges considering gradient sparsity prior. Authors proposed a method that decomposes the reflection-contaminated image into two images such that the total number of corners and edges is minimized. However, this method performs poorly as the complexity in the images texture rises. Levin et al. [7] rely on user assistance to simplify the problem. Although this method successfully manages to separate the reflections from a single image to a certain degree, manually marking the image for the presence of reflection is difficult and is only practical for a small number of images. Shih et al. [8] reduce the ill-posedness by the use of ghosting cues and exploit the Gaussian mixture model (GMM) to learn image priors. Ghosting cues are the double reflections shifted by some distance, arising due to light being reflected at both the surfaces of a glass pane. Ghosting cues arise mostly in case of double pane glass or if the glass is quite thick, so this method works only on a small subset of images containing reflection. Wan et al. [9] assume prior that the background layer contains sharp and well-defined edges and the reflection layer is relatively smoother and use this relative difference in the smoothness as a cue to create a depth of field (DoF) confidence map, which then is used to classify edges as part of either the background layer or the reflection layer. This method cannot remove reflection from images with tiny textures or small reflection artefacts.

Learning-Based Approaches

Recent works have leveraged deep learning capabilities to solve the reflection removal problem.

Fan et al. [10] follow the same prior assumption as [9], i.e. reflection layer is off-focus and blurry. They created a synthetic dataset that mimics the assumed prior and proposed a two-stage cascaded network. The first stage predicts the edges of the background layer, and the predicted edges are used by the second layer to guide the background layer recovery. Wan et al. [11] improved [10] two stages into a single end-to-end concurrent network to predict the edges and separate the layers. Zhang et al. [12] combined three losses (feature loss, adversarial loss and exclusion loss) to train the proposed end-to-end network. The network and the losses are tuned to exploit both low-level and high-level information; still, this method performs poorly on images with high exposure. Recently GAN (Generative Adversarial Networks)-based methods [13, 14] have yielded good results, but still have issues handling images with extreme exposures, and [13] produces fails to produce images with natural colours as the colour tone is altered when the parts of reflection appear in the background. Also, the problem inherent with GANs is their complexity, both in terms of network architecture and the time and parameter tuning required to train the network. Some other related articles [15–20], and proposed deep learning-based IoT methods to solve different problems.

3 Proposed Method

Training Dataset

All the existing learning-based single image reflection removal methods fail to fully take advantage of their respective proposed models due to the lack of labelled training data. Lack of labelled training data is a common problem in computer vision community and though there are some workarounds even they are limited in cases in which they can be applied. The most common workaround is creating a synthetic dataset. The problem with creating a synthetic dataset is that they usually fail to truly mimic the wide range and variety of classes present in natural datasets, which in turn limits the capabilities of the method to deal with naturally occurring images. Another workaround is assuming priors and proposing a method considering the priors. Priors usually restrict the scope of the approach by setting some bounds on the input and thereby making the approach more tailored towards dealing with inputs from the smaller range. Such methods may or may not perform equivalently for inputs outside this range.

We have followed the first approach, i.e. to expand the dataset using synthetic images. We have used data from multiple sources to accomplish this and merged it with images from already available datasets to train a reflection removal model.

PASCAL Visual Object Classes (VOC) dataset [15] is used to create synthetic images with reflection artefacts. To synthesize one image, two images are selected from the dataset and cropped into 256 × 256-sized patches. Then, one patch is selected as the background, and the other one as the reflection. Both the images are merged using the following equation:

Fig. 8.3 Image triplet (B, R, I) from the training dataset

$$\boldsymbol{I} = \boldsymbol{\alpha} * \boldsymbol{B} + \boldsymbol{\beta} * (\boldsymbol{G} \otimes \mathbf{R}) \tag{8.2}$$

where
I*, *B and ***R*** are n × m × 3 matrices representing the resulting synthetic image, the background patch, and the reflection patch, respectively
***α*:** blending weight for the background patch
***β*:** blending weight for the reflection patch and $\boldsymbol{\beta} = (\mathbf{1} - \boldsymbol{\alpha})$
***G*:** represents the Gaussian blur operation applied on reflection patch

Reflection patch is blurred out using Gaussian blur, and then blending weight $\alpha \in [0.6, 0.8]$ is used to combine both the images. The generated dataset contains 50,000 synthetic images. An image triplet (containing the background, the reflection and the final blended result) from the training dataset can be seen in Fig. 8.3.

Model Description (Table 8.1)

Loss Function

Loss value is a measure of how off the predictions are from true values. Loss function reflects the performance of the model and provides a quantitative measure of accuracy. The loss function is a key aspect in determining how good a solution the trained model is as the objective function being realized while in training phase is to minimize the loss. Therefore, the loss function must be chosen in a way that minimization of the loss value results in the model predicting value close to the true values, and for this to happen the loss function must be tailored to the problem being solved. As images are at the centre of reflection removal problem, we will first

Table 8.1 Model architecture

Layer (type)	Connected to	# Filters	Kernel size	Activation	Output shape
conv2d_1(Conv2D)	conv2d[0][0]	64	9 × 9	ReLU	256 × 256 × 64
conv2d_2 (Conv2D)	conv2d_1[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_3 (Conv2D)	conv2d_2[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_4 (Conv2D)	conv2d_3[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_5 (Conv2D)	conv2d_4[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_6 (Conv2D)	conv2d_5[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_7 (Conv2D)	conv2d_6[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_8 (Conv2D)	conv2d_7[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_9 (Conv2D)	conv2d_8[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_10 (Conv2D)	conv2d_9[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_11 (Conv2D)	conv2d_10[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_12 (Conv2D)	conv2d_11[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_13 (Conv2D)	conv2d_12[0][0]	64	5 × 5	ReLU	256 × 256 × 64
tf_op_layer_add (TensorFlowOpLayer)	conv2d_9[0][0], conv2d_13[0][0]	–	–	ReLU	256 × 256 × 64
conv2d_14 (Conv2D)	tf_op_layer_add [0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_15 (Conv2D)	conv2d_14[0][0]	64	5 × 5	ReLU	256 × 256 × 64
tf_op_layer_add_1 (TensorFlowOpLayer)	conv2d_7[0][0], conv2d_15[0][0]	–	–	ReLU	256 × 256 × 64
conv2d_16 (Conv2D)	tf_op_layer_add_1 [0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_17 (Conv2D)	conv2d_16[0][0]	64	5 × 5	ReLU	256 × 256 × 64
tf_op_layer_sub (TensorFlowOpLayer)	conv2d_5[0][0], conv2d_17[0][0]	–	–	ReLU	256 × 256 × 64
conv2d_18 (Conv2D)	tf_op_layer_sub[0] [0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_19 (Conv2D)	conv2d_18[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_20 (Conv2D)	conv2d_19[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_21 (Conv2D)	conv2d_20[0][0]	64	5 × 5	ReLU	256 × 256 × 64
conv2d_22 (Conv2D)	conv2d_21[0][0]	64	9 × 9	ReLU	256 × 256 × 64
conv2d_23 (Conv2D)	conv2d_22[0][0]	3	9 × 9	ReLU	256 × 256 × 3

Padding = "same"
Strides = (1,1)
Total params: 2,744,131
Trainable params: 2,744,131
Non-trainable params: 0

take a metric that can precisely measure the similarities between two images and use it inside the loss function to calculate the loss value.

Structural similarity (SSIM) index provides the quantitative measure of structural similarity between images and is formulated on a similar basis using which the human visual system assess the similarity between two scenes. Our visual system has

Fig. 8.4 Original image, increased contrast, blurred image

evolved to extract structural information from the scene; therefore, calculating the structural resemblance between two images can provide a decent approximation of actual similarity between them.

SSIM provides better similarity estimation than other measures for images as every pixel is weighted equally in case of peak signal-to-noise ratio (PSNR) and mean squared error (MSE), irrespective of the fact that any change in its value will be noticeable to the human observer or not. This could lead high variations in MSE and PSNR scores for the image pairs when the contrast or brightness changes in one of the image, even though these modifications don't have a significant effect on human observer assessing image similarity as can be seen in Fig. 3.2. Therefore, structural similarity index is more likely to find such image pairs more similar, as the structural information in the image pair would resemble closely, as the SSIM index is calculated on various windows of an image (Fig. 8.4).

SSIM index ranges from 0 to 1, 0 meaning that the images share no structural similarity and 1 meaning perfect structural similarity between images, which is only possible for identical images. Three components, namely, luminance, contrast and structure, are used in the process of calculating SSIM index for two perfectly aligned images of same size $\boldsymbol{x}$ and $\boldsymbol{y}$.

Luminance comparison $\boldsymbol{l}(\boldsymbol{x},\boldsymbol{y})$ is given by:

$$\boldsymbol{l}(\boldsymbol{x},\boldsymbol{y}) = \frac{2\boldsymbol{\mu}_x\boldsymbol{\mu}_y + \boldsymbol{C}_1}{\boldsymbol{\mu}_x^2 + \boldsymbol{\mu}_y^2 + \boldsymbol{C}_1} \tag{8.3}$$

Contrast comparison $\boldsymbol{c}(\boldsymbol{x},\boldsymbol{y})$ is given by:

$$\boldsymbol{c}(\boldsymbol{x},\boldsymbol{y}) = \frac{2\boldsymbol{\sigma}_x\boldsymbol{\sigma}_y + \boldsymbol{C}_2}{\boldsymbol{\sigma}_x^2 + \boldsymbol{\sigma}_y^2 + \boldsymbol{C}_2} \tag{8.4}$$

Structure comparison $\boldsymbol{s}(\boldsymbol{x},\boldsymbol{y})$ is given by:

$$\boldsymbol{s}(\boldsymbol{x},\boldsymbol{y}) = \frac{\boldsymbol{\sigma}_{xy} + \boldsymbol{C}_3}{\boldsymbol{\sigma}_x\boldsymbol{\sigma}_y + \boldsymbol{C}_3} \tag{8.5}$$

where
$\boldsymbol{\mu}_x$ is the average of intensities of $\boldsymbol{x}$, $\boldsymbol{\mu}_y$ is the average of intensities of $\boldsymbol{y}$,

$\boldsymbol{\sigma_x^2}$ is the variance of intensities of $\boldsymbol{x}$, $\boldsymbol{\sigma_y^2}$ is the variance of intensities of $\boldsymbol{y}$,
$\boldsymbol{\sigma_{xy}}$ is the covariance of intensities of $\boldsymbol{x}$ and $\boldsymbol{y}$, and
$\boldsymbol{C_1}$, $\boldsymbol{C_2}$ and $\boldsymbol{C_3}$ are used to avoid instability when denominators are close to zero.
$C_1 = (K_1\,L)^2$, $C_2 = (K_2\,L)^2$, $C_3 = C_2/2$,
$K_1 << 1$ and $K_2 << 1$, and L is the dynamic range.

Using the above-mentioned three components, SSIM index is calculated as follows:

$$\mathbf{SSIM}(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{l}(\boldsymbol{x}, \boldsymbol{y}) \cdot \boldsymbol{c}(\boldsymbol{x}, \boldsymbol{y}) \cdot \boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}) \tag{8.6}$$

Substituting the values of $l(x,y)$, $c(x,y)$ and $s(x,y)$ in the above equation, we get

$$\mathbf{SSIM}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\left(2\boldsymbol{\mu_x}\boldsymbol{\mu_y} + \boldsymbol{C_1}\right)\left(2\boldsymbol{\sigma_{xy}} + \boldsymbol{C_2}\right)}{\left(\boldsymbol{\mu_x^2} + \boldsymbol{\mu_y^2} + \boldsymbol{C_1}\right)\left(\boldsymbol{\sigma_x^2} + \boldsymbol{\sigma_y^2} + \boldsymbol{C_2}\right)} \tag{8.7}$$

This can be converted into loss function to calculate the loss between the estimated background layer and the actual background layer as follows:

$$\mathbf{loss_{SSIM}}\left(\boldsymbol{y_{true}}, \boldsymbol{y_{pred}}\right) = \mathbf{1} - \mathbf{SSIM}\left(\boldsymbol{y_{true}}, \boldsymbol{y_{pred}}\right) \tag{8.8}$$

4 Experiment and Results

In this section, we present the details of the experiments performed and their evaluation. Detailed discussion on the impact of various parameters of the proposed approach on the overall performance is also included.

Training Details

Trained the network with the following parameters:

Number of epochs: 65
Batch size: 32
Validation split: 0.2
Shuffle: True
Optimizer: Adam ($\alpha = 0.0001$, $\beta 1 = 0.9$, $\beta 2 = 0.999$)
Loss function: MSE, loss_SSIM

A combination of MSE and our custom loss function based on SSIM index has been used during training. The process of training was carried out in two phases: in

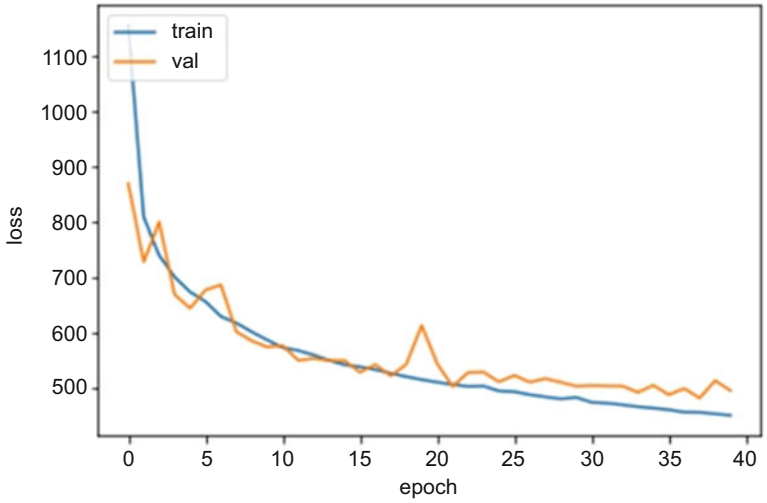

Fig. 8.5 Training and validation loss vs epoch graph – Phase I

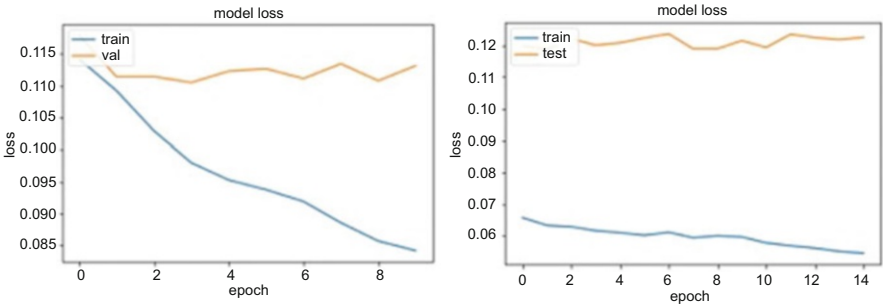

Fig. 8.6 Training and validation loss vs epoch graph – Phase II

the first phase, we have used the entire training dataset with MSE as the loss function and trained the network for 40 epochs, and, in the second phase, the network was trained for a total of 25 epochs on smaller subsets of the training dataset with SSIM-based loss function (Figs. 8.5 and 8.6).

Experimental Set-Up

Experiments were carried out on the system with the following configurations:

CPU: Intel Xeon Silver 4114
Memory: 64 GB DDR4
GPU: NVIDIA Quadro P5000

GPU Memory: 16 GB GDDR5X
Storage: 4 TB
Operating system: Ubuntu 18.04.4 LTS (Bionic Beaver)

Deep learning libraries used: Keras with TensorFlow backend, TensorFlow 2.1.0, CUDA 10.1, cuDNN 7.6.

Programming language and major libraries used: Python 3.6, NumPy, OpenCV, Matplotlib.

Performance Evaluation Metrics

For performance evaluation, the most common metrics in comparing two images are PSNR value and SSIM score (refer to Sect. 3.3 for a detailed description of SSIM index). Peak signal-to-noise ratio (PSNR) is the ratio between a signal's maximum power and the power of corrupting noise that affects the quality of images and videos. Generally, PSNR is conveyed on a logarithmic decibel scale. The formula for PSNR between the original image and the noisy image is given in the following equation:

$$\text{PSNR} = 20 * \log_{10} \frac{max_f}{\sqrt{\text{MSE}}} \tag{8.9}$$

where
max_{f} – maximum signal value present in the original image

Mean squared error (MSE) is

$$\text{MSE} = \frac{1}{mn} \sum_{0}^{m-1} \sum_{0}^{n-1} \| \text{f}(i,j) - \text{g}(i,j) \|^2 \tag{8.10}$$

where
f – original image in matrix form
g – predicted image in matrix form
m – number of rows in input images
n – number of columns in input images
i, j – co-ordinates of a current pixel location in input images

Testing Dataset

Benchmarking SIR2 dataset [16] with images containing real scenes is used to assess the performance and capabilities of the trained network. SIR2 dataset is released by

Table 8.2 Comparison of average SSIM scores for proposed and competing methods

Method	SSIM
LB14 [2]	0.793
AY07 [7]	0.834
SK15 [8]	0.785
WS16 [9]	0.862
FAN17 [10]	0.854
XR18 [12]	0.823
Proposed (*after Phase I*)	0.7855682
Proposed (*after Phase II*)	0.81121135

Table 8.3 Comparison of average PSNR values for proposed and competing methods

Method	PSNR
LB14 [2]	21.735
AY07 [7]	21.436
XR18 [12]	20.28
Proposed (after Phase I)	15.966344
Proposed (after Phase II)	16.880268

Rapid-Rich Object Search (ROSE) Lab, NTU, Singapore. It has a large number of diverse images containing a reflection, along with the corresponding ground truth of their reflection and background layers. It contains both indoor (controlled) scenes and outdoor (wild) scenes. Indoor scenes include postcards and solid objects used in day-to-day life, such as fruits, toys, mugs, etc. Outdoor scenes contain real-world entities, such as trees, gardens, cars, buildings, etc. with varying illuminations, scales and distances. SIR2 dataset contains a total of 500 image triplets with 200 triplets each for postcard dataset and solid object dataset and 100 triplets for wild scene dataset (Table 8.2 and 8.3 and Fig. 8.7).

5 Conclusion and Future Work

In this dissertation, we have studied the single image reflection removal problem and proposed a method to suppress the reflection and recover the background layer. Our approach focuses mainly on using simple network architecture along with a loss function tailored to the demands of the problem. To address the issue of lack of labelled training data, we have created and used synthetic dataset for training our network.

Experimental results validate the efficacy and efficiency of our approach. A similar approach can be used in solving the problems, such as super resolution, where the current approaches use complex network architectures, including autoencoders and generative adversarial networks.

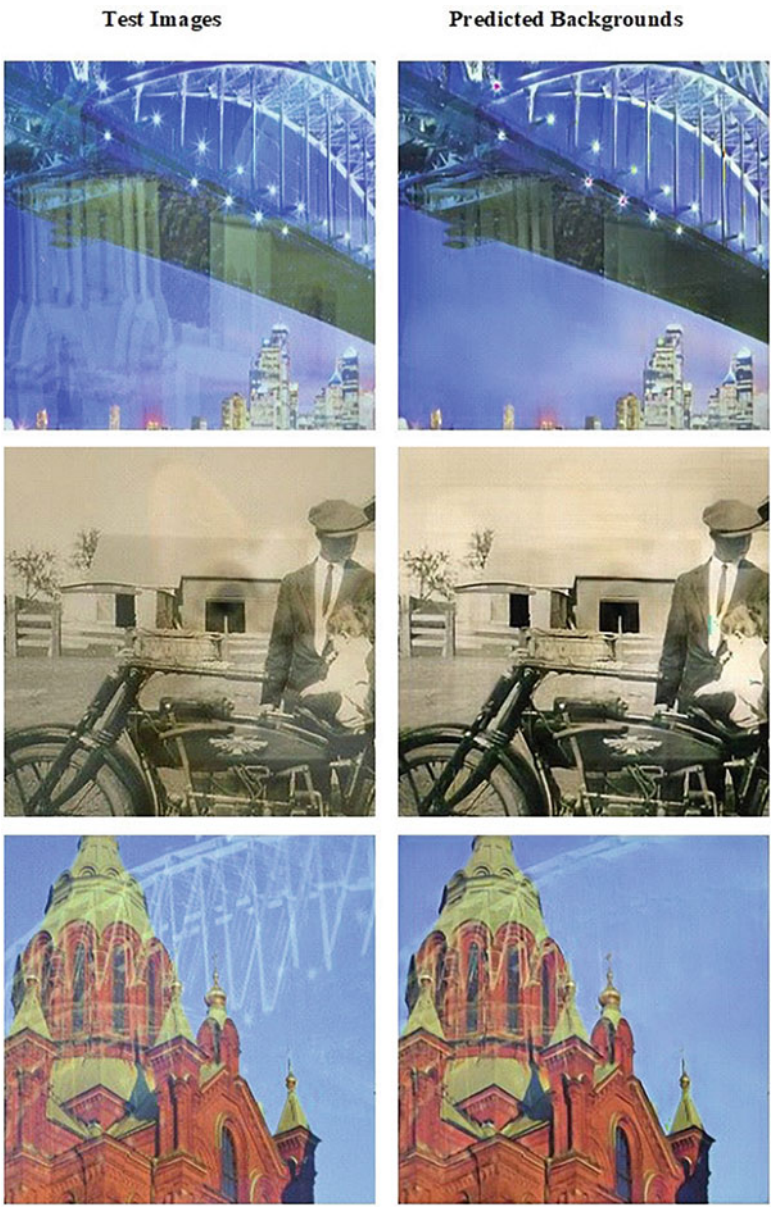

Fig. 8.7 Reflection removal results on test images by the proposed model

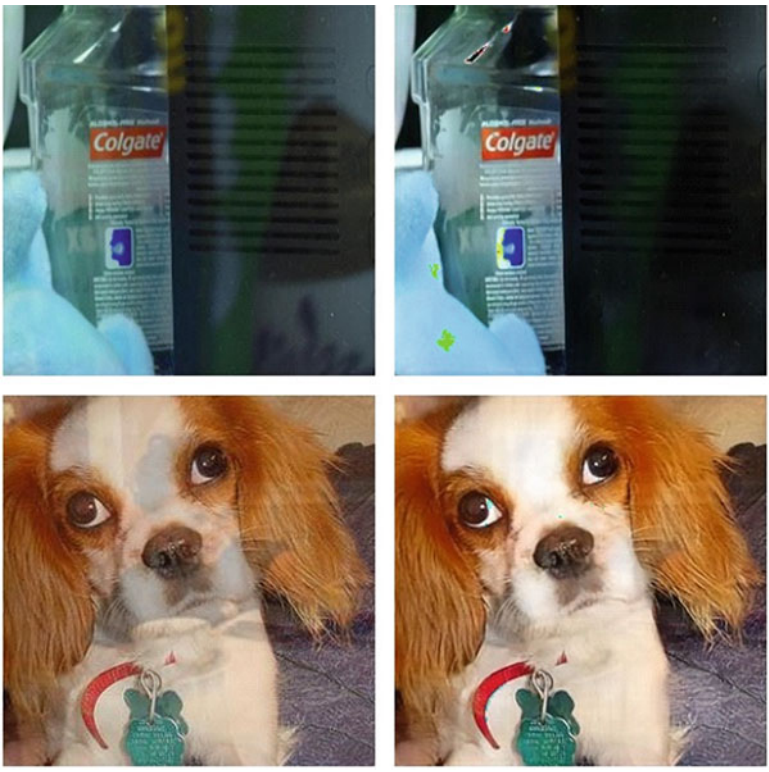

Fig. 8.7 (continued)

Our method has produced decent results but still fails to outperform the state-of-the-art method. Future works can focus on using the ground truth of the reflection layer in addition to the ground truth of the background layer to further improve the effectiveness of the approach.

References

1. Wei, K., Yang, J., Fu, Y., Wipf, D., Huang, H.: Single image reflection removal exploiting misaligned training data and network enhancements. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8170–8179 (2019). https://doi.org/10.1109/CVPR.2019.00837
2. Abiko, R., Ikehara, M.: Single image reflection removal based on gan with gradient constraint. IEEE Access **7**, 148790–148799 (2019). https://doi.org/10.1109/ACCESS.2019.2947266. Author, F., Author, S.: Title of a proceedings paper. In: Editor, F., Editor, S. (eds.) CONFERENCE 2016, LNCS, vol. 9999, pp. 1–13. Springer, Heidelberg (2016)
3. Veringham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010)

4. Wan, R., Shi, B., Duan, L.-Y., Tan, A.-H., Kot, A.C.: Benchmarking single-image reflection removal algorithms. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 3942–3950 (2017). https://doi.org/10.1109/ICCV.2017.423
5. Gopikrishnan, S., Priakanth, P., Srivastava, G.: DEDC: sustainable data communication for cognitive radio sensors in the internet of things. Sustain. Comput. Inf. Syst. **29**, 100471 (2021). https://doi.org/10.1016/j.suscom.2020.100471
6. Vallathan, G., John, A., Thirumalai, C., Mohan, S., Srivastava, G., Lin, J.C.-W.: Suspicious activity detection using deep learning in secure assisted living IoT environments. J. Supercomput. **77**(4), 3242–3260 (2021). https://doi.org/10.1007/s11227-020-03387-8
7. Guo, T., Yu, K., Srivastava, G., Wei, W., Guo, L., Xiong, N.N.: Latent discriminative low-rank projection for visual dimension reduction in green internet of things. IEEE Trans. Green Commun. Netw. **5**(2), 737–749 (2021). https://doi.org/10.1109/TGCN.2021.3062972
8. Zhu, D., Sun, Y., Du, H., Cao, N., Baker, T., Srivastava, G.: HUNA: a method of hierarchical unsupervised network alignment for iot. IEEE Internet Things J. **8**(5), 3201–3210 (2021). https://doi.org/10.1109/JIOT.2020.3020951
9. Guo, X., Cao, X., Ma, Y.: Robust separation of reflection from multiple images. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2195–2202 (2014). https://doi.org/10.1109/CVPR.2014.281
10. Li, Y., Brown, M.S.: Single image layer separation using relative smoothness. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2752–2759 (2014). https://doi.org/10.1109/CVPR.2014.346
11. Agrawal, A., Raskar, R., Nayar, S., Li, Y.: Removing photography artifacts using gradient projection and flash-exposure sampling. ACM Trans. Graph. **24**, 828–835 (2005). https://doi.org/10.1145/1186822.1073269
12. Schechner, Y.Y., Shamir, J., Kiryati, N.: Polarization and statistical analysis of scenes containing a semireflector. J. Optic. Soc. Am. A. **17**(2), 276–284 (2000). https://doi.org/10.1364/JOSAA.17.000276
13. Kong, N., Tai, Y.-W., Shin, J.S.: A physically-based approach to reflection separation: from physical modeling to constrained optimization. IEEE Trans. Pattern Anal. Mach. Intell. **36**(2), 209–221 (2014). https://doi.org/10.1109/TPAMI.2013.45
14. Levin, A., Zomet, A., Weiss, Y.: Separating reflections from a single image using local features. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004, vol. 1 (2004). https://doi.org/10.1109/CVPR.2004.1315047
15. Levin, A., Weiss, Y.: User assisted separation of reflections from a single image using a sparsity prior. In: Pajdla, T., Matas, J. (eds.) Computer Vision – ECCV 2004, pp. 602–613. Springer, Berlin/Heidelberg (2004)
16. Shih, Y., Krishnan, D., Durand, F., Freeman, W.T.: Reflection removal using ghosting cues. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3193–3201 (2015). https://doi.org/10.1109/CVPR.2015.7298939
17. Wan, R., Shi, B., Hwee, T.A., Kot, A.C.: Depth of field guided reflection removal. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 21–25 (2016). https://doi.org/10.1109/ICIP.2016.7532311
18. Fan, Q., Yang, J., Hua, G., Chen, B., Wipf, D.: A generic deep architecture for single image reflection removal and image smoothing. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 3258–3267 (2017). https://doi.org/10.1109/ICCV.2017.351
19. Wan, R., Shi, B., Duan, L.-Y., Tan, A.-H., Kot, A.C.: CRRN: multi-scale guided concurrent reflection removal network. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4777–4785 (2018). https://doi.org/10.1109/CVPR.2018.00502
20. Zhang, X., Ng, R., Chen, Q.: Single image reflection separation with perceptual losses. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4786–4794 (2018). https://doi.org/10.1109/CVPR.2018.00503