UKRAINIAN CATHOLIC UNIVERSITY

BACHELOR THESIS

Reflection Removal with Generative Adversarial Networks

Author:
Volodymyr ZABULSKYI

Supervisor:
Jiri MATAS

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv, Prague 2020

Declaration of Authorship

I, Volodymyr ZABULSKYI, declare that this thesis titled, "Reflection Removal with Generative Adversarial Networks" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"There is an art, it says, or rather, a knack to flying. The knack lies in learning how to throw yourself at the ground and miss."

Douglas Adams, Life, the Universe and Everything

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Bachelor of Science

Reflection Removal with Generative Adversarial Networks

by Volodymyr ZABULSKYI

Abstract

We propose an approach for the single image reflection removal problem. Our model is based on a feature pyramid network (FPN), trained with adversarial and perceptual losses. Additionally, we address the problem of bright spots removal when only a small portion of an image is covered with the reflection. The difficulty of collecting real-world data makes the problem even harder to solve. We propose a novel method of collecting real-world data, that does not require any additional devices but a camera, and is cheaper than the existing ones. We collected a small dataset with this approach.

Acknowledgements

I want to express my sincere gratitude to Jiri Matas for supervising this research project and providing valuable feedback. Special thanks to all my friends from the Center for Machine Perception (CMP) and Ukrainian Catholic University for constant encouragement. Thanks to all managers for not trying to call the police while I was collecting data in their facilities.

Contents

List of Figures

List of Tables

List of Abbreviations

SIRR	**S**ingle **I**mage **R**eflection **R**emoval
CNN	**C**onvolutional **N**eural **N**etwork
DNN	**D**eep **N**eural **N**etwork
GAN	**G**enerative **A**dversarial **N**etwork
FPN	**F**eature **P**yramid **N**etwork
ReLU	**Re**ctified **L**inear **U**nit
MSE	**M**ean **S**quared **E**rror
MAE	**M**ean **A**bsolute **E**rror
SSIM	**S**tructural **Sim**ilarity
PSNR	**P**eak **S**ignal-to-**N**oise **R**atio
SOTA	**S**tate-**O**f-**T**he-**A**rt

Chapter 1

Introduction

Photos, taken from the inside of a cafe, car, or a shopping mall, may be contaminated with an undesirable glass reflection. Although sometimes the reflection may be part of the picture concept, it usually corrupts the image and thus is unpleasant. One may want to obtain a clean background image by removing the reflection from the photo. Restoring the reflection-free background layer - reflection removal - is an active research topic in computer vision.

Although we can intuitively understand what a reflection is and imagine how a clean background should look like, one may have a hard time mathematically defining what a "reflection" is. To reduce unclarity, we assume that statistics of the reflection and the background layers are profoundly different - their edges and textures should rarely overlap.

Commonly, we formulate reflection removal tasks as a layer separation. Let $I \in \mathbb{R}^{n \times m \times 3}$ be the picture containing reflection. One can approximate I as $I = B + R$, where $B \in \mathbb{R}^{n \times m \times 3}$ is the reflection-free background image, and $R \in \mathbb{R}^{n \times m \times 3}$ is the reflection image. The goal is to recover the background layer B, given image I. Without introducing any constraints or using any prior knowledge, the problem is severely ill-posed, thus challenging.

Many recent works made attempts to leverage the layers statistics by introducing additional images, such as video sequence [Wen et al., 2019] or flash-no flash image pairs [Chang et al., 2020]. Nevertheless, solving the problem with a single image is still challenging. Some use prior assumptions and handcrafted features [Yang et al., 2019], thus are not robust to outlining changes in edges, colors, and textures. Other methods rely on hardware, extracting extra layers, such as near-infrared light and depth layer [Li and Lun, 2019].

Recently, it became common to use convolutional neural networks (CNN) to extract sophisticated features of the images. Attempts to retrieve reflection-free background using deep neural networks (DNN) were made [Chang et al., 2020; Fan et al., 2019; Zhang, Ng, and Chen, 2018], showing remarkable results in a wide range of images. In our experiments, we use state-of-the-art architectures, such as generative adversarial networks (GANs) [Goodfellow, Bengio, and Courville, 2016], proved to be effective in related image restoration tasks, to solve the reflection removal problem as a layer separation task.

Neural nets are extremely efficient in obtaining high-level statistics from data, which makes their performance highly dependent on the quality of the data, see Figure 1.1. At best, the training data should be from the same domain as the data from the problem we try to solve - in our case, both from the real world. Given the expensiveness and difficulty of obtaining real-world data for reflection removal, most of the approaches rely on synthetic data. As the experiments show, methods trained on such data poorly generalize to the real-world data, which makes the performance of every approach highly limited to the training data. Some of the approaches [Zhang,

FIGURE 1.1: This figure shows examples of the models trained on synthetic data comparing synthetic test images and the real-world. Although they work well on the synthetic data, they are highly limited in performance in the wild. At best, not detecting the reflection and doing nothing significant to the picture, at worst, corrupting the image.

Ng, and Chen, 2018] used both the synthetic dataset and a real "mini" dataset to train their models. Although they perform well on the similar real-world pictures as used during training (the same glass, similar scenes, the same camera), they still fail on arbitrary real-world images.

Several works [Zhang, Ng, and Chen, 2018; Wan et al., 2017; Han and Sim, 2019] proposed methods for obtaining real-world data. In those works, authors used a piece of glass, taking images with and without it, obtaining corrupted and reflection-free ground truth pairs. In [Wan et al., 2017] and [Han and Sim, 2019], they also placed an opaque black sheet behind the glass plane to get the ground truth reflection as well, forming triplets data samples. Partly, in [Wan et al., 2017], the authors used postcards as a background and the reflection, leveraging the headache on working with a big piece of glass, but limiting the dataset to a few different image samples.

These approaches have some apparent limitations. First, one must physically have a piece of glass and a place to put it, which limits the data diversity, as one can't remove the glass in any arbitrary cafe or a shopping mall. Next, the background scene must be static, as the background changes during the time they put off the glass can introduce severe noise to the data. We propose a data collection approach, partly free of the mentioned limitations. Although having other constraints and challenges, we believe it to be cheaper and more straightforward than the mentioned ones.

Chapter 2

Related Work

2.1 Neural Networks

Neural networks are biologically inspired algorithms, aiming to mimic the way humans learn. Many applications use feed-forward neural networks, designed to take a fixed-size input (for example, biometric data) and give a fixed-size output (for example, the probability of a heart attack). They usually consist of several layers. Layers that are neither output nor input layers are called hidden layers. Neural nets train, using the backpropagation algorithm, which updates the weights inside layers, concerning the current prediction error. Algorithms, trained to extract useful properties and features of the data are called unsupervised learning algorithms. Training with data, where each example is associated with its label, is called supervised learning. Neural networks are particularly useful to extract non-trivial patterns, especially when the problem is too hard to solve with handcrafted features.

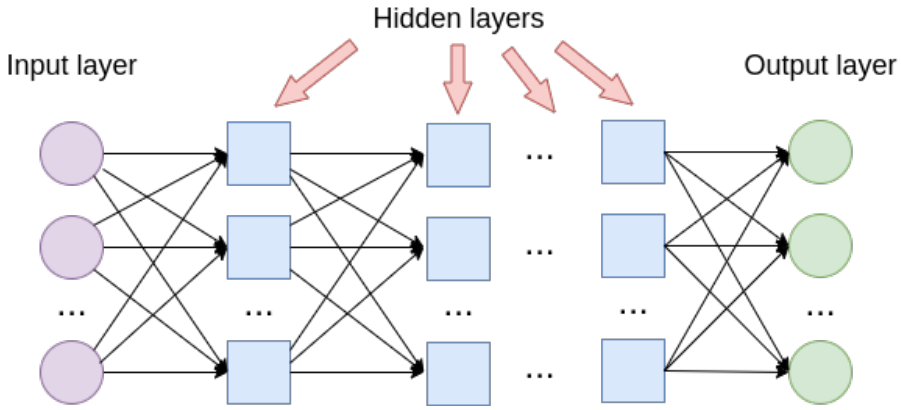

FIGURE 2.1: Schematic picture of a neural net.

2.2 Convolutional Neural Networks

Convolutional Neural Networks [Lecun et al., 1998], or CNNs are algorithms that aim to work with grid-like data, such as images. The foundation of CNNs lies in the convolution operation; that is, convolutional networks are neural networks that use convolution in place of general matrix multiplication in at least one of their layers [Goodfellow, Bengio, and Courville, 2016].

Beside LeNet (Figure 2.2) [Lecun et al., 1998], there are few more CNN architectures worth mentioning. AlexNet [Krizhevsky, Sutskever, and Hinton, 2012] had a great success in expanding CNNs to a larger scale. As the authors pointed out in their paper: "We trained one of the largest convolutional neural networks to date

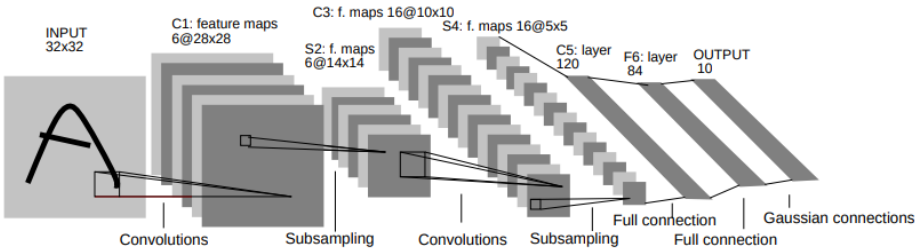

FIGURE 2.2: LeNet [Lecun et al., 1998] - a classic CNN architecture. Created for the character recognition in the '90s.

on the subsets of ImageNet." They were also one of the first to successfully implement nonlinear activation function - Rectified Linear Unit (ReLU). VGG [Simonyan and Zisserman, 2014] has made a foundation for the next generation of "very deep" Convolutional Networks with a "very small (3×3) convolution filters", following the tradition of using ReLUs from AlexNet. Feature maps from the VGG network are widely used as the perceptual loss in the image reconstruction tasks [Johnson, Alahi, and Li, 2016]. InceptionNet architecture [Szegedy et al., 2015] uses filers of different sizes (1×1), (3×3) and (5×5) to concatenate them in so-called "inception blocks", allowing extracting features at different scales, keeping the computation cost unchanged. ResNet [He et al., 2016] introduced "Shortcut Connections," which act as networks inside a network, skipping one or more layers before computing the output. They mainly addressed the solution to the problem of vanishing gradient, when repeated multiplications diminish the value of the final gradient.

2.3 Generative Adversarial Networks

GANs are a particular type of network constructed to learn the distribution of the data. They consist of two main parts: the Generator, which learns the data distribution, and the Discriminator, trained to evaluate how precise the estimate of the Generator is. They play minimax game in which the Generator tries to fool the Discriminator with its generated data, and Discriminator tries to learn a function that maximizes the distance between the generator's fake data and the real one [Goodfellow, Bengio, and Courville, 2016]. Mathematically, the game can be expressed with the following formula:

$$min_G max_D V(D,G) = \mathbb{E}_{\boldsymbol{x} \sim p_{data}(\boldsymbol{x})}[logD(\boldsymbol{x})] + \mathbb{E}_{z \sim p_z(z)}[log(1 - D(G(\boldsymbol{z})))] \quad (2.1)$$

where $V(D,G)$ is the value function of the minimax game, p_{data} is the data distribution, and $p_{\boldsymbol{z}}$ is the model distribution.

2.4 Feature Pyramid Network

The Feature Pyramid Networks (FPNs) are general-purpose and originally were used for object detection [Lin et al., 2016]. They are composed of bottom-up and top-down pathways. The bottom-up pathway is a regular backbone convolutional network (e.g., [Simonyan and Zisserman, 2014; Szegedy et al., 2015, He et al., 2016]),

containing semantic information at different scales and serves for feature extraction. The top-down pathway hallucinates higher spatial resolution features. Feature maps of the same spatial size from bottom-up and top-down pathways are merged via lateral connections.

2.5 Reflection Removal

In this section, we first review the general approaches for the Single Image Reflection Removal (SSIR) [Jin, Süsstrunk, and Favaro, 2018; Zhang, Ng, and Chen, 2018; Wen et al., 2019; Li et al., 2019; Fan et al., 2019; Sun et al., 2019] and also those who use some priors or constraints [Springer and Weiss, 2017; Zhang et al., 2019; Kim, Huo, and Yoon, 2019; Chang et al., 2020; Liu et al., 2019; Wan et al., 2019]. Then, we discuss the problem of generating synthetic data and attempts to make it more realistic, as well as attempts to collect a reasonable amount the real-world data.

Many methods for image reflection removal approaches have been proposed, based on different principles. Some attempts try to solve the most general task of the reflection removal and take as input a single image only [Jin, Süsstrunk, and Favaro, 2018; Zhang, Ng, and Chen, 2018; Wang, Li, and Yang, 2018; Wen et al., 2019; Li et al., 2019; Fan et al., 2019; Sun et al., 2019].

Deep models were widely used to target the reflection removal problem. [Fan et al., 2017] present a Cascaded Edge and Image Learning Network (CEILNet) that can be used to solve different image restoration tasks such as layer separation (e.g., reflection removal) and image filtering (e.g., image smoothing). Perceptual loss [Johnson, Alahi, and Li, 2016], which compares high-level features of the resulting images, was successfully applied to improve the performance of the algorithms by [Zhang, Ng, and Chen, 2018], along with adversarial loss, used to address unrealistic color degradation and undesirable subtle residuals. [Fan et al., 2019] demonstrated a novel training approach, while during the training, the model is fed with image pairs, and is able to remove reflections only from a single input for evaluation.

Other approaches include unsupervised methods [Gandelsman, Shocher, and Irani, 2018; Liu and Lu, 2019], those, which estimate image priors, such as depth [Sun et al., 2019], physics of light [Kim, Huo, and Yoon, 2019], object semantics [Liu et al., 2019]. Some require user interaction to guide the algorithms [Zhang et al., 2019; Springer and Weiss, 2017], hardware to work with flash-no flash image pairs [Chang et al., 2020]. Some address more specific problems, such as face reconstruction from reflection [Wan et al., 2019].

In [Liu et al., 2020], authors use motion image pairs to get a better estimate of the background behind the glass, plus also successfully building their model to remove small obstructions. In [Alayrac et al., 2019], authors apply a controllable model, able to adjust attention for the two layers, recovering whether reflection or background, from a video sequence.

The general task of SIRR is still challenging, despite some impressive progress in recent research. The issue lays on the foundation of the assumption that the reflection and the background layer have very different statistics, e.g., edges of both layers rarely overlap, textures rarely coincide. For instance, they make and the assumption that the reflection layer is blurred, as it is usually out of the camera focus. These do not always take place in real-world images.

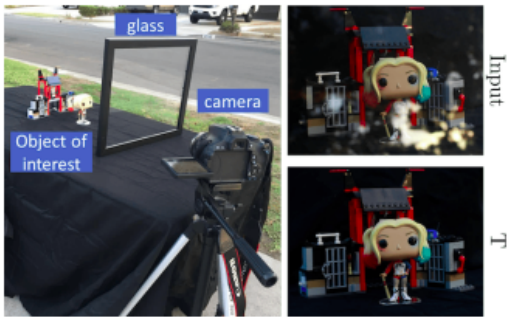

FIGURE 2.3: Real data collection setup and captured images by [Zhang, Ng, and Chen, 2018]. They capture two images with and without the glass with same camera settings in a static scene. Right column from top to bottom: captured image with reflection and the ground-truth transmission image T.

2.5.1 Real World Data

Collecting real-world data is expensive; thus, no reasonably big real dataset exists. Such a dataset might be collected, using a piece of glass, taking two images with and without it with fixed camera parameters. This approach is limited to only static scenes (difficult to collect pictures with people, clouds, animals, grass), with reasonable noise otherwise. Moreover, the refraction effect of the glass usually produces some noise. [Zhang, Ng, and Chen, 2018] collected a dataset with 110 image pairs: image with the glass and its corresponding ground-truth reflection-free image (see examples on the Figure 2.3).

Collecting the ground-truth reflection layer is challenging too. One may put a black, opaque object, behind the glass and obtain the reflection. [Wan et al., 2017] made a collection SIR^2 (Single Image Reflection Removal) of three different datasets with a total of 500 image triplets. Postcard Dataset, where both reflection and transmission are postcards (200 samples), Solid Object Dataset, taken in lab conditions with static objects such as toys, mugs, fruits (200 samples), and Wild Dataset, taken outside with natural environment illumination (100 samples). Although the Postcard dataset contains 200 image triplets, it should be noted that the authors used only 5 different postcards (combining them to obtain 20 combinations), plus using 10 different conditions, thus obtaining 200 different, but not diverse triplets. They used types of glass with different thickness (3mm, 5mm and 10mm). They used seven different aperture sizes and choose seven different exposure times, corresponding to the seven aperture settings to make the brightness of each picture approximately constant [Wan et al., 2017]. The same is true for the Solid dataset, which contains 20 unique samples, plus 10 different conditions, resulting in 200 samples. The ground-truth reflection is rarely a sharp image as it is usually out of camera focus. Some examples of SIR^2 are in Figure 2.4.

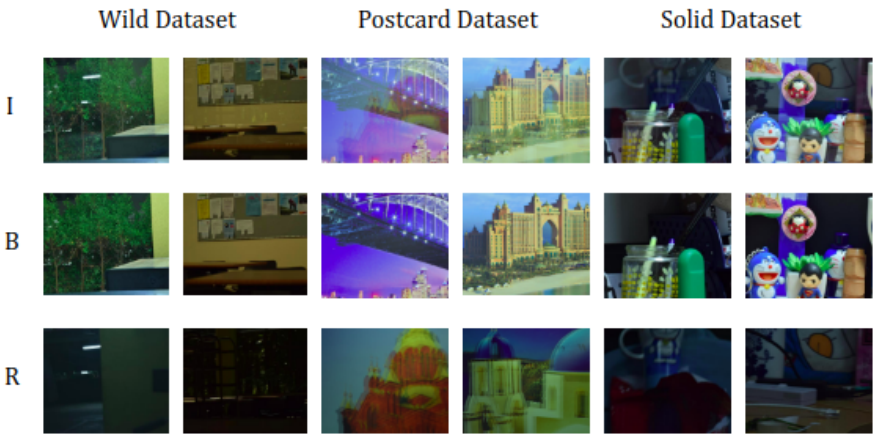

FIGURE 2.4: Real-world data triplets collected by [Wan et al., 2017]. From top to bottom rows: I - images with reflection, B - ground-truth background images, R - ground-truth reflection images.

2.5.2 Synthetic Data

As a physical phenomenon, reflection light always adds up to the transmission light in a formed image (Figure 2.5). Thus, reflection synthesis is an additive process, where we add a reflection layer to the background layer, with other processing. Trivially, the blended image can be computed as a linear combination of corresponding pixels of a background layer and reflection layer: $I = B + \alpha R$. However, images synthesized in this manner rarely look realistically; thus, some processing is required.

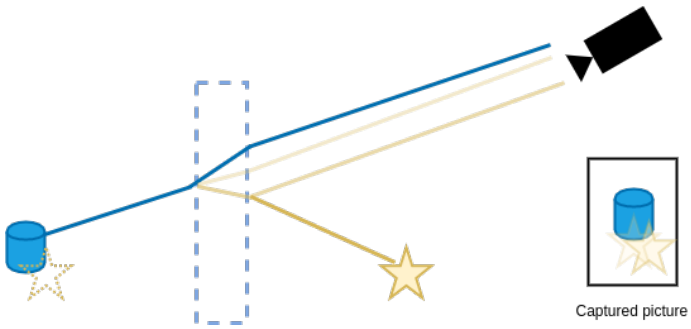

FIGURE 2.5: Physical model of image formation. Here the light from the reflected object (star) reflects twice, projecting with different intensities and different locations. This effect is mostly visible with a thick glass. For simplicity, lens models are omitted in the scheme. Sometimes reflection objects are out of the lens focus. causing additional reflection blurriness. Figure inspired by [Wan et al., 2017].

In [Zhang, Ng, and Chen, 2018] authors propose several processing steps to obtain closer images to those we observe in real life. They collected a dataset of 13700 image pairs from Flickr [*Flickr* 2020] ether one outdoor, one indoor, or vice versa. The visualization of the blending approach is shown in Figure 2.6.

Steps to reproduce (these details are extracted from the official implementation of the [Zhang, Ng, and Chen, 2018] paper):

1. Pick any two images for the reflection (R) and the background (B), normalized to $[0,\ 1]$

2. Apply Gaussian blur on the reflection:

   $R_b = G(R^{2.2},\ k \times k,\ \omega)$, where
   $\omega \in [1;\ 5]$
   $k = \lfloor 2 \times \lceil 2 \times \omega \rceil + 1 \rfloor = kernel\ size$

3. Apply random darkness for each channel separately

   $R_{bd} = R_b - (\mu - 1) \times \alpha_1$, where
   $\alpha_1 \in [1.08;\ 1.18]$
   $\mu = max(1,\ mean_{\{R,G,B\}}(\{R_b + T^{2.2}\} > 1))$

4. Apply Gaussian mask M with a random center to simulate vignette

   $R_{bdm} = R_{bd} \times M$

5. Obtain blended Image with $\alpha_2 \in [0.8; 1]$

   $I = \sqrt[2.2]{R_{bdm} + B^{2.2} \times \alpha_2}$

6. Clip* $\{I > 1\} := 1; \{I < 0\} := 0$

   * The reflection image usually gets dark enough so that clipping rarely gives any effect

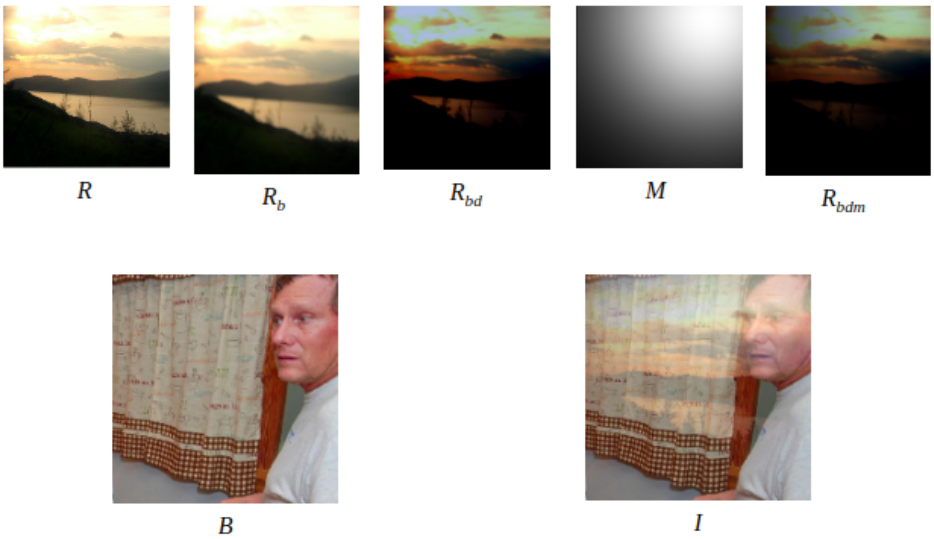

FIGURE 2.6: Synthesis of images with reflection by [Zhang, Ng, and Chen, 2018]. For the notation details, please refer to the numeric list above.

Name	Real	Has GT reflection	Outdoor	# of samples
Zhang synth	-	+		13700
Zhang real	+	-	mixed	110
SIR^2 Postcard	+	+	-	200
SIR^2 Solid	+	+	-	200
SIR^2 Wild	+	+	+	100

TABLE 2.1: Summary of the reflection removal datasets used in this work.

Chapter 3

Proposed Approach

The main goal of our research is to create an efficient pipeline to separate the background layer of the image from the reflection layer. Given the image I, we want to get the background B without any prior knowledge about the nature of the reflection R. Also, we do experiments on the problem of bright spots removal, where reflection occupies roughly 10% of the image (e.g., lamp reflection), for which we created a real-world dataset.

3.1 Data

3.1.1 Collecting Real-World Data

This section describes our approach to acquiring real-world data. The source image is the photo from which the reflection should be removed; the reference image is a supporting photo to extract the ground-truth behind the bright spot; the recovered image is the estimated ground-truth.

The background is assumed to be static and laying on one plane. The source image is taken with the reflection present; the reference image is taken by a slight side movement of the camera, such that the background behind the reflection reveals. However the bright spots are not necessary to disappear from the reference image completely. Ideally, one should use a stereo camera, taking both the source and the reference images at once, assuming the sufficient distance between two cameras. Then, taking similar key points from both images, the homography of the background of the images is estimated. The reference image is transformed to match the source image's background homography. Taking into account that reflection is guaranteed to add illumination to the background, the pixel-wise minimum of both images gives us the estimation of the ground-truth image.

The challenges we faced while obtaining the data:

- Obstacles between background plane and window (trees, people, plants), which can cause poor homography estimation (Figure 3.4).
- The background is not a perfect plane, especially noticeable if it's located relatively close to the camera (Figure 3.4, row 2).
- Algorithm could wrongly select parts of a reflection as the key points (possible to overcome with multiple runs) (Figure 3.4, row 3).
- Bright spots overlap in source and transformed reference image if the camera was not moved enough to the side (Figure 3.4, row 4).
- Non-static background, such as clouds, nature, people, vehicles (Figure 3.4, row 5).

- Glass reflections from windows, belonging to the background are located on a very different plane (although we don't want to remove or change them).
- One has to walk into facilities and ask managers for permission to "take a picture of your window because I'm a scientist and need it for my research." I am genuinely grateful to all the waiters and waitresses who didn't kick me out and didn't call the police while suspiciously looking at what I am doing in their restaurant.

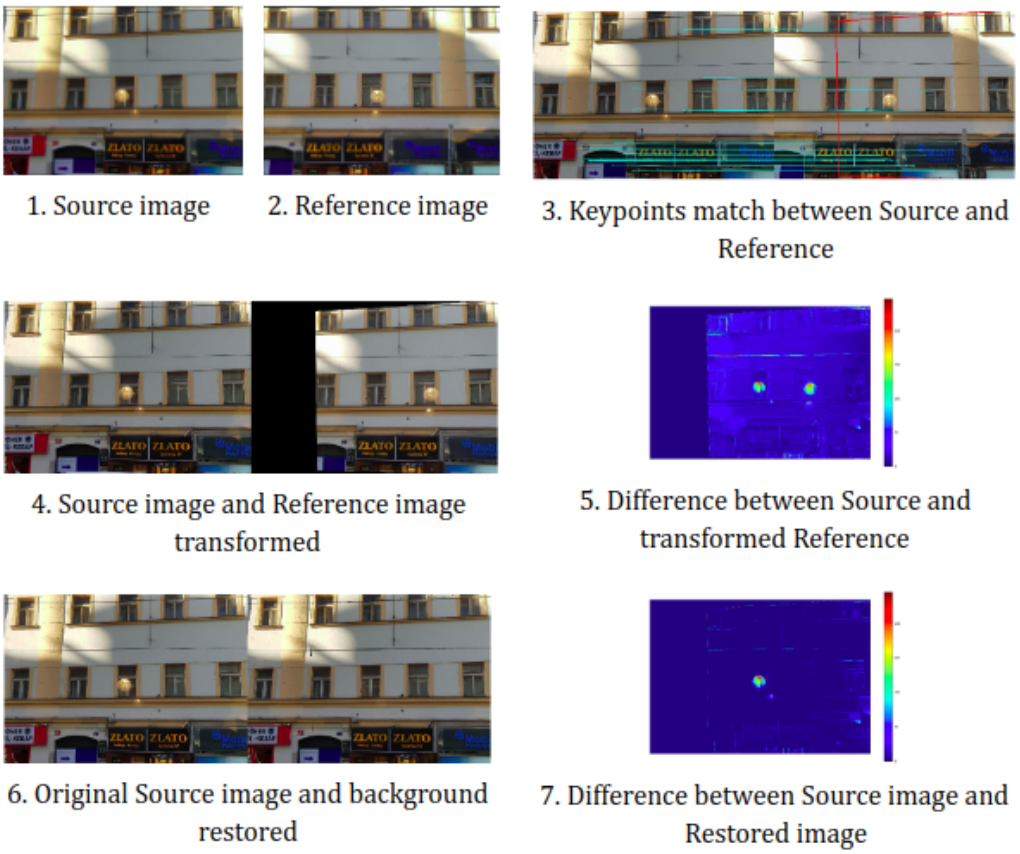

FIGURE 3.1: Recovering images, using homography estimation. Picture 3 shows matched keypoints, connected with blue strings, and the plane that corresponds to the source image's patch to be recovered, with the red quadrangle. For the visualization, we reduced the number of strings to 100. In this case, the real number was 2000. Picture 5 shows the absolute pixel-wise difference of the source image and transformed reference image. Blue color notes a small difference, where red and yellow shows a big difference. At best, in picture 7, all the pixels, but the reflection part should be blue. Because most of the pixels outside the reflection are blue, we consider the recovery to be successful

FIGURE 3.2: Real data examples collected using our approach. The source image is taken with the reflection present; the reference image is taken by a slight side movement of the camera, such that the background behind the reflection reveals. The background is restored by taking a pixel-wise minimum from the source image and transformed reference image.

FIGURE 3.3: This Figure follows Figure 3.2. Real data examples collected using our approach. The second sample was taken by opening an automatic glass door, the last one by switching off the lamp. Although the camera remained static in the last two examples, reference image transformation was still applied, as it showed better accuracy on the pixel level.

1. Trees, the fence, and the hill are located on different planes, leading to failed homography estimation. Here the key points were selected from the fence; thus it's the least damaged from the "recovering" .

2. The background building is located too close to the camera, diminishing the assumption of the planar background.

3. As the reflection occupies too much space, the algorithm chose the bright spots as the key points, leading to a false homography estimation.

4. The movement of the camera to the right was not enough to fully reveal the GT background; thus part of a bright spot is present on the recovered image.

5. Kafka's head moved during the procedure, breaking the assumption that the background is static

FIGURE 3.4: Different failures during the recovering process of the reflection-free images. The last column shows the absolute pixel-wise difference between the source and the recovered images.

3.1.2 Bright Spots Data Synthesis

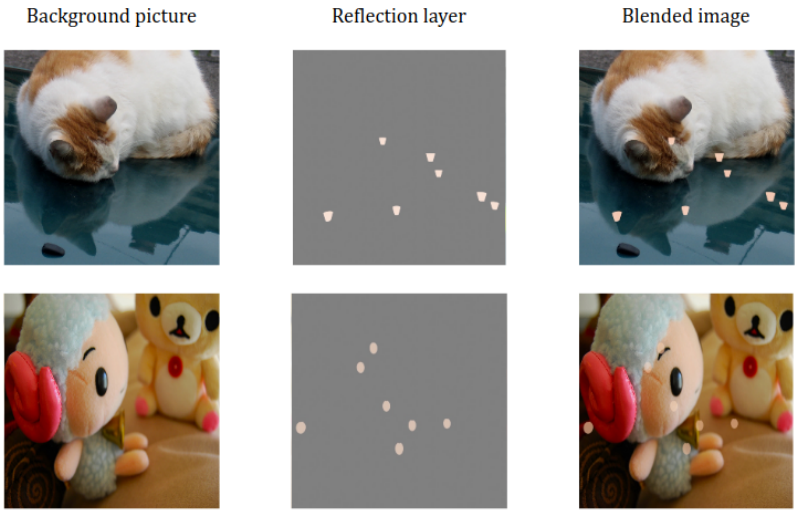

FIGURE 3.5: Synthetic images with bright spots reflection, generated with Blender [Community, 2018].

Aiming to simulate real-world indoor light, we generated data using Blender [Community, 2018] glass reflection simulation algorithm, which calculates a linear combination of background pictures and reflection layer. We also add minor light noise to the blended image. Background pictures are from [Lin et al., 2014] dataset. Examples are shown on the Figure 3.5.

3.2 Network Architecture

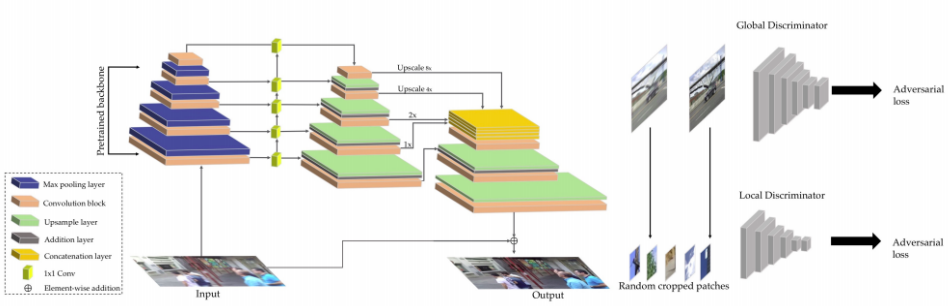

FIGURE 3.6: DeblurGAN-v2 pipeline architecture [Kupyn et al., 2019]. The setting consists of a feature pyramid network (FPN) in the encoder-decoder style. The input image is added to the output image to make the learning focus on the residual.

We use DeblurGAN-v2 [Kupyn et al., 2019] architecture in our model: Figure 3.6, which is proven to work well on the similar image restoration domain - deblurring. The setting consists of a feature pyramid network (FPN) in the encoder-decoder style, where the input shape is equal to the output shape. The encoder backbone contains features at different levels; we output five of them to be upsampled and concatenated with the corresponding decoder layers. Additionally, two upsampling

and convolutional layers are added afterward to restore the original image size. We add the input to the output image to make the learning focus on the residual. Inception-ResNet-v2 [Szegedy et al., 2017] is used for the backbone, although one can freely replace it with any other efficient network.

3.3 Loss Functions

Perceptual loss As proposed in [Kupyn et al., 2019] we use perceptual loss (L_{percep}), calculating the Euclidean distance between VGG19 [Simonyan and Zisserman, 2014] conv_3_3 feature maps. In the experiments we additionally propose to replace single feature map of VGG19 with weighted sum of multiple: conv_1_2, conv_2_2, conv_3_2, conv_4_2, and conv_5_2, as proposed in [Zhang, Ng, and Chen, 2018]. However, later evaluation showed weaker performance of such loss. We also selected mean-square-error (MSE) loss (L_{mse}) in advantage to other "classical" L_1 distance (MAE).

Adversarial loss PatchGAN, as proposed in [Isola et al., 2016], operates on image patches 70×70, and is proven to make sharper results, than a standard global discriminator. To take advantage of both global and patch discriminators, we use both in our complete loss.

The overall loss is formulated as the following:

$$L = \lambda_{adv} \times L_{adv} + \lambda_{percep} \times L_{percep} + \lambda_{mse} \times L_{mse} \tag{3.1}$$

Where λ notes weights to balance between losses importance.

In the initial experiments, we use the same loss function as proposed by [Kupyn et al., 2019]. For this specific setup, perceptual loss used for reflection removal by [Zhang, Ng, and Chen, 2018] showed lower performance than the perceptual loss used for deblurring by [Kupyn et al., 2019], according to our evaluation.

3.3.1 Patched loss

In the bright spots removal problem, most of the pixels are unaffected, or infinitesimally affected by the reflection. Taking this into account, one can crop the bright spots on an image on several patches. Given that the reflection-free areas should be unchanged by the network, one may want to "amplify" the loss function to the areas with bright spots. While having synthetic data with the ground-truth reflection layer, localization and cropping bright spots or computing bright spots mask is a trivial task. Patched loss is computed as an average loss of each bright spots patch, weighted and added to the "global" loss. Steps to compute the Patched loss are the following:

1. Given ground truth reflection layer, compute bright spots centroids, using convolutions
2. Crop each bright spot by the fix-sized square (in our experiments, we empirically set the size to (32×32))
3. Calculate the average loss for all patches
4. Multiply by coefficients and add to the "global" loss function

Chapter 4

Experiments

4.1 Metrics

Intuitively, we want to measure the statistics of the two images to conclude how perceptually close they are to each other. Structural similarity (SSIM) index and Peak signal-to-noise ratio (PSNR) are standard metrics in this domain. Although it is not trivial to formalize the sense of human perception, and they both are not the best perception criteria, they are widely used to compare approaches to the state-of-the-art.

SSIM index considers changes in structural information, luminance and contrast between image patches x and y [Wang et al., 2004]:

$$SSIM(x,y) = [l(x,y)]^{\alpha} \times [c(x,y)]^{\beta} \times [s(x,y)]^{\gamma}, \; where \tag{4.1}$$

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}, \tag{4.2}$$

$$c(x,y) = \frac{\sigma_x\sigma_y}{\sigma_x^2 + \sigma_y^2 + C_2}, \tag{4.3}$$

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}, \tag{4.4}$$

where μ, σ and σ_{xy} are the mean, standard deviation and covariance of the image patches respectively, $C_1, C_2, C_3 > 0$ are stabilizing constants to prevent the division when the denominator is approaching zero, and $\alpha, \beta, \gamma > 0$ are weights to control the importance of the three factors. The final value is calculated as a weighted average index of patches, taken by a sliding window. SSIM reaches the maximum value of 1 for two identical images and 0 for images with no structural similarity.

PSNR, given images x and y, computes with the following formula:

$$PSNR(x,y) = 10\, log_{10} \frac{MAX_x^2}{\sqrt{MSE}} \tag{4.5}$$

where MAX_x is the maximum possible intensity value and MSE stands for the mean-square-error between two images of shape $M \times N$:

$$MSE(x,y) = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} (x_{ij} - y_{ij})^2 \tag{4.6}$$

PSNR is measured in decibels (dB), and MAX_x is 255 for images, encoded with 8 bits. For color images (e.g., RGB), the value is the same, except that MSE is the sum over all squared value differences additionally divided by three. The higher is PSNR, the higher is the similarity between two images.

4.2 Training and implementation details

All of the models are implemented using PyTorch [*PyTorch*]. The model was trained on NVIDIA GeForce GTX 1080 Ti. The images were reshaped to 256×256. We used Adam optimizer [Kingma and Ba, 2014]. The learning rate for the generator is 10^{-4} and remains unchanged. After 50 epochs, we linearly decrease the learning rate to 10^{-7}. We train with the batch size 1. The SSIM index is used from the skimage [Walt et al., 2014] library.

4.3 SIRR

In [Kupyn et al., 2019], authors propose to measure the perceptual loss by calculating the Euclidean distance between VGG19 [Simonyan and Zisserman, 2014] *conv_3_3* feature maps to solve the deblurring problem. In contrast, [Zhang, Ng, and Chen, 2018] suggests the distance between the weighted sum of several VGG19 feature maps (*conv_1_2*, *conv_2_2*, *conv_3_2*, *conv_4_2*, and *conv_5_2*) to extract features at different scales and solve the reflection removal problem. We experimented with replacing default perceptual loss in the DeblurGAN-v2 model with loss proposed by [Zhang, Ng, and Chen, 2018]. However, our evaluation showed weaker performance of such setup, see Table 4.1.

loss		train		validation	
λ_{adv}	L_{percep}	SSIM	PSNR	SSIM	PSNR
0.001	Kupyn	**0.91**	**27.35**	**0.909**	**26.85**
0	Kupyn	0.9	26.9	0.906	26.17
0.001	Zhang	0.891	26.5	0.904	26.53
0	Zhang	0.895	26.47	0.875	25.83

TABLE 4.1: Choosing perceptual loss between [Zhang, Ng, and Chen, 2018] and [Kupyn et al., 2019], trained on synthetic data by [Zhang, Ng, and Chen, 2018]. λ_{adv} is weighting the Adversarial loss value. Best results are bold.

As shown in Table 4.2, it is hard and non-trivial on some metrics to outperform even a baseline approach with particular real-world data.

Loss	CEILNet		SIR^2 Wild		SIR^2 Solid		SIR^2 Postcard	
$L_{percep}, \lambda_{adv}$	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR
Baseline	0.75	14.0	**0.898**	19.02	**0.88**	23.62	0.87	20.93
Kupyn, 0.001	0.778	15.534	0.865	23.721	0.878	23.17	**0.894**	22.225
Kupyn, 0	**0.786**	15.865	0.863	**23.907**	**0.88**	23.44	**0.895**	22.988
Zhang, 0.001	0.769	15.582	0.857	23.793	0.875	**23.688**	0.880	**23.332**
Zhang., 0	0.778	**16.014**	0.845	23.614	0.856	23.186	0.878	23.328

TABLE 4.2: Choosing perceptual loss between [Zhang, Ng, and Chen, 2018] and [Kupyn et al., 2019] on real data with the same setup as in [4.1]. Additionally, we provide a baseline, where the target image is the same as the input image. Best results are bold, results worse than the baseline are red.

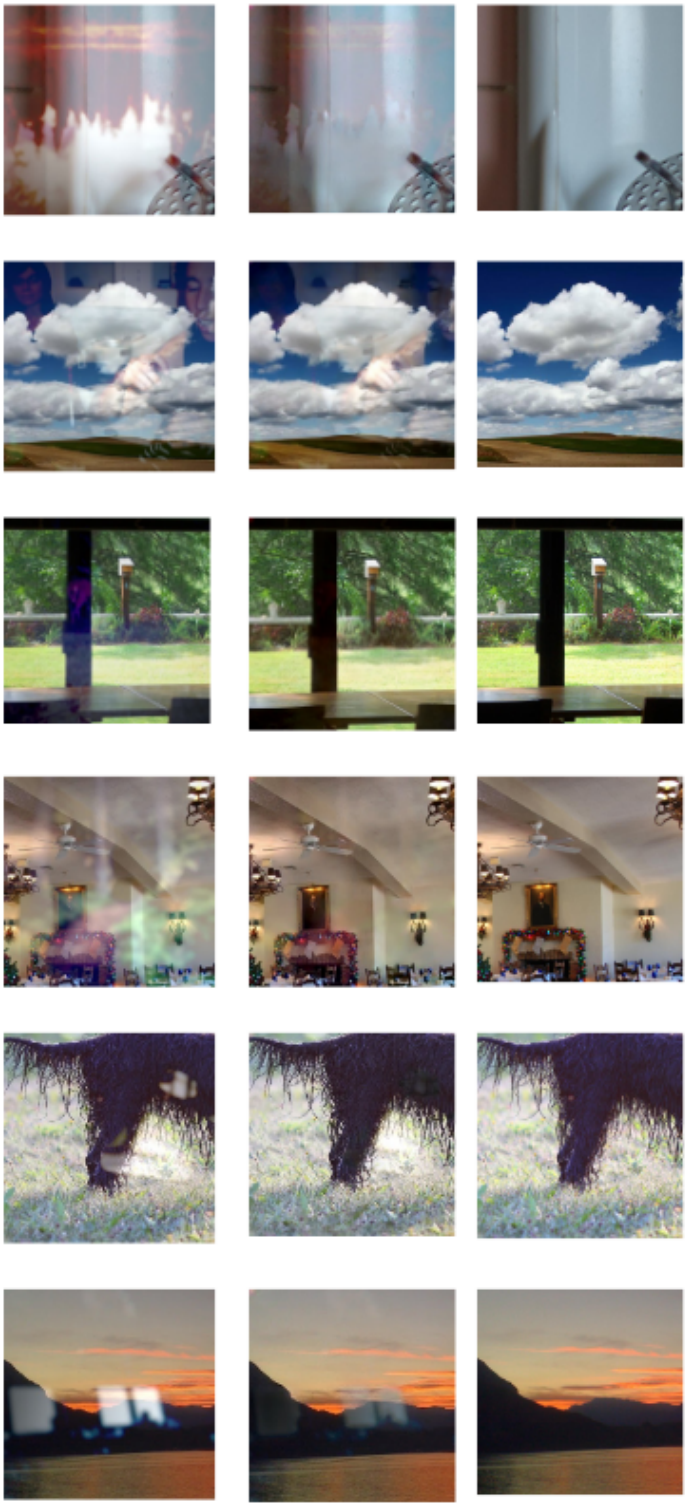

FIGURE 4.1: Examples of our model performing on synthetic data.
Order: input image, predicted image, ground-truth image

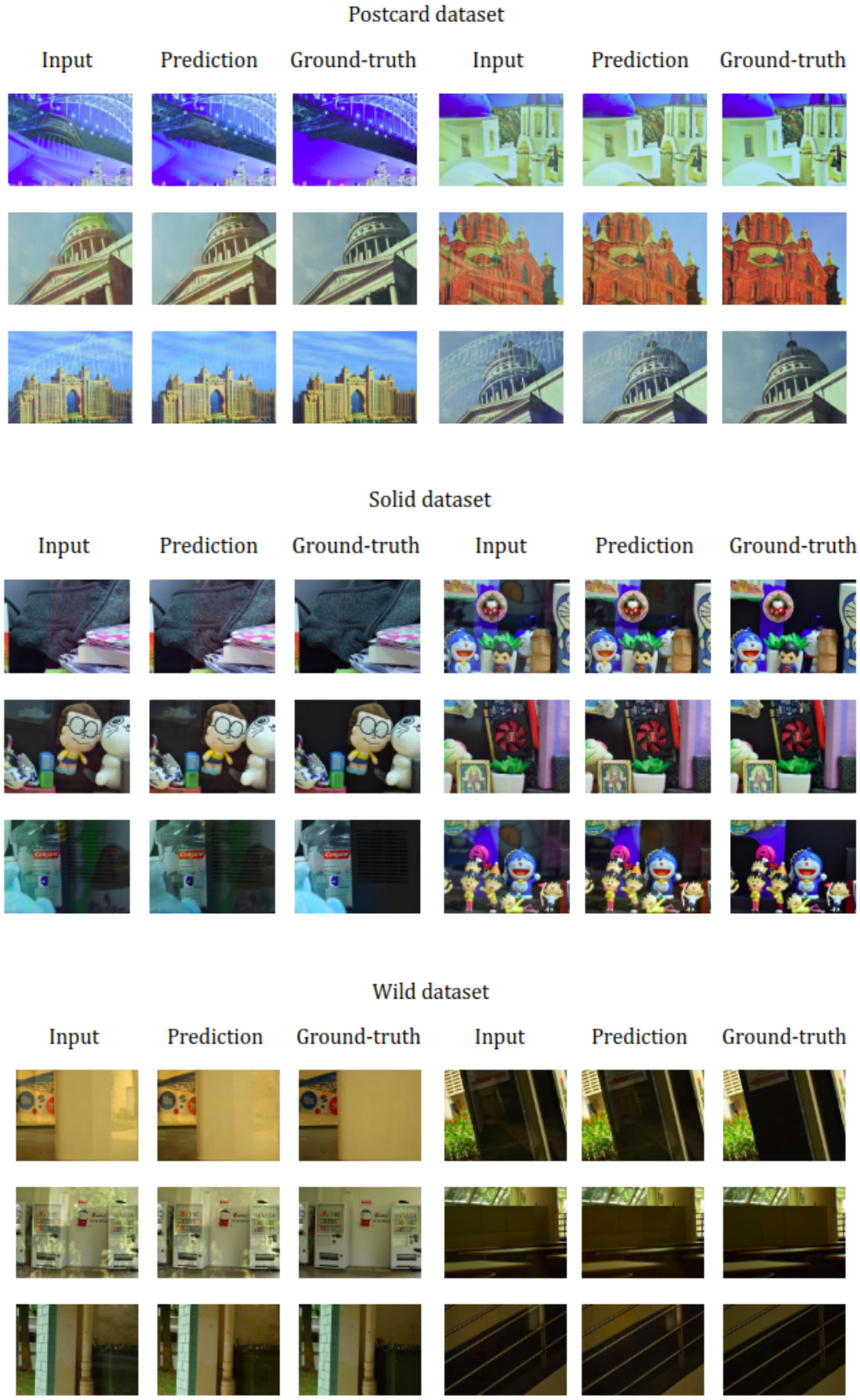

FIGURE 4.2: Examples of our model performing on SIR^2 real data.
Order: input image, predicted image, ground-truth image.

From the examples in Figure 4.2, we assume that in most of the cases, the model managed to detect the reflection layer and not mess up the background. Our further experiments stress this assumption by providing the model with the ground-truth reflection layer mask in the bright spots removal problem. Table 4.3 shows performance of our approach on SIR^2 dataset

Methods	$SIR^2 Total$		$SIR^2 Postcard$		$SIR^2 Solid$		$SIR^2 Wild$	
	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR
Baseline	0.88	22.72	0.87	20.93	**0.88**	**23.62**	**0.891**	**25.89**
Zhang	0.837	19.23	0.797	15.80	**0.88**	22.14	0.831	21.15
Ours	**0.884**	**23.3**	**0.895**	**22.988**	**0.88**	23.44	0.863	23.91
Relative	+5.6%	+21.1%	+12.2%	+45.4%	0	+5.8%	+3.8%	+13%

TABLE 4.3: Comparing our model with model by [Zhang, Ng, and Chen, 2018] trained on the same data. Best results are bold, results worse than the baseline are red. Baseline measures accuracy when the input and the output images are the same. Relative change for ours vs Zhang.

4.4 Bright Spots Removal

Standard setup Knowing that the model should not change most of the pixels in data instances, we observed inefficiency of SSIM and PSNR metrics, as returning the input image gives nearly the same score as the maximum possible. For these experiments, the visual evaluation was done. Objective evaluation for this problem is still an open question. Figure 4.3 shows the results with the same model setup, as in earlier experiments.

Four channel input To address the hypothesis of the model successfully detecting the reflection layer, we introduce a four-channel input to the network. The first three channels are normal RGB channels, and the 4'th channel is a binary mask of the reflection, where pixels have a value of 1 when belonging to the strong reflection patch and 0 otherwise. The first layer of weights of the pre-trained network is copied to a 4'th input layer.

Varying Patched Loss weights Overall, the patched loss has the same formula as the "global loss," but computed for patches. For the faster search, MSE and Perceptual loss share the same weight. Worth to mention that by introducing such loss, the model sometimes produces artifacts in the reflection-free part of an image, see Figure 4.6.

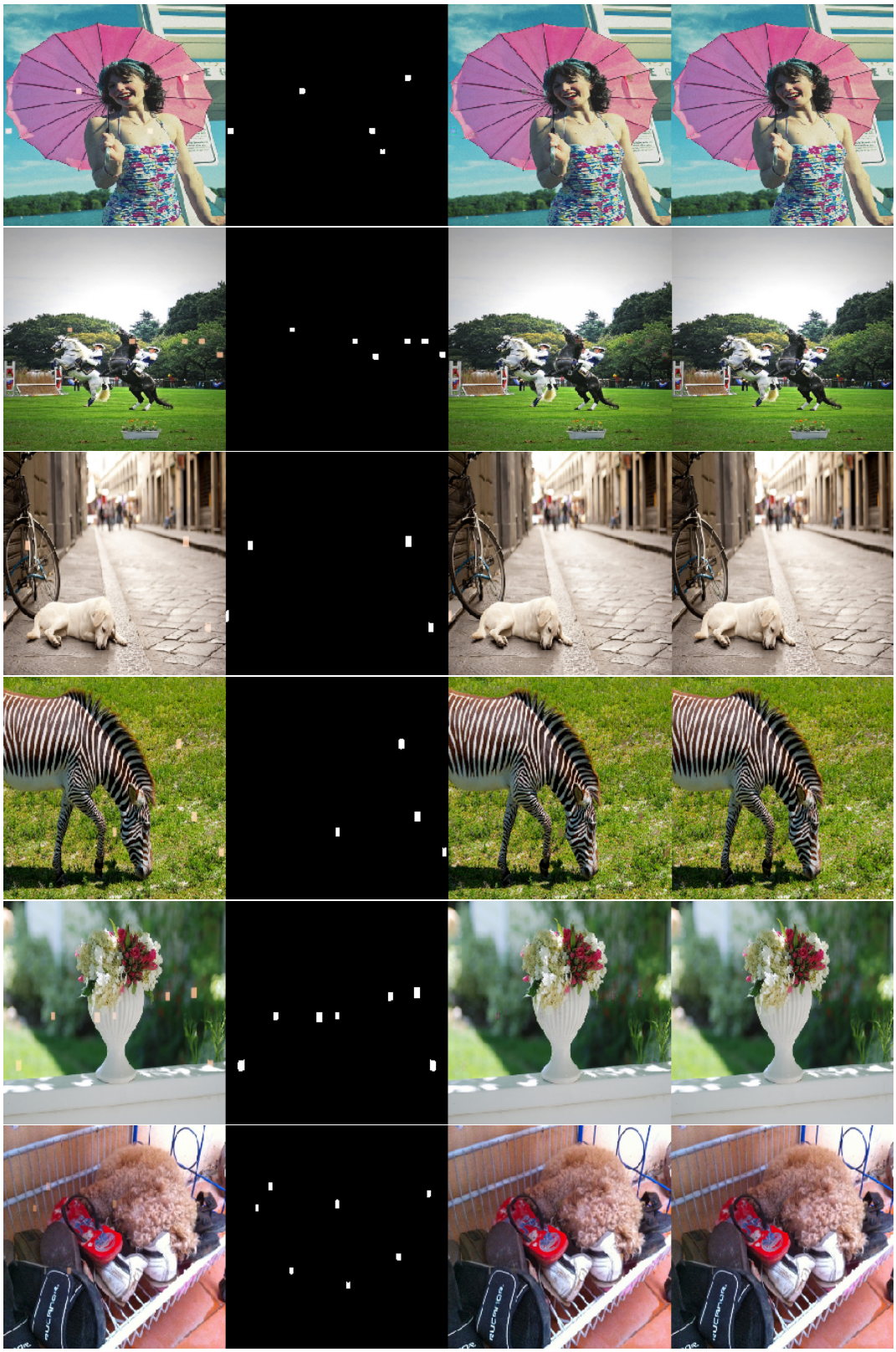

FIGURE 4.3: Bright Spots Removal with global loss function. From left to right: input image, bright spots mask, prediction, ground-truth. The mask is provided for visualization purpose only and is not used for training in this setup. The last row shows example of the input without any reflection, confirming that the model is able to detect and not damage the background layer. Zoom in for a better view.

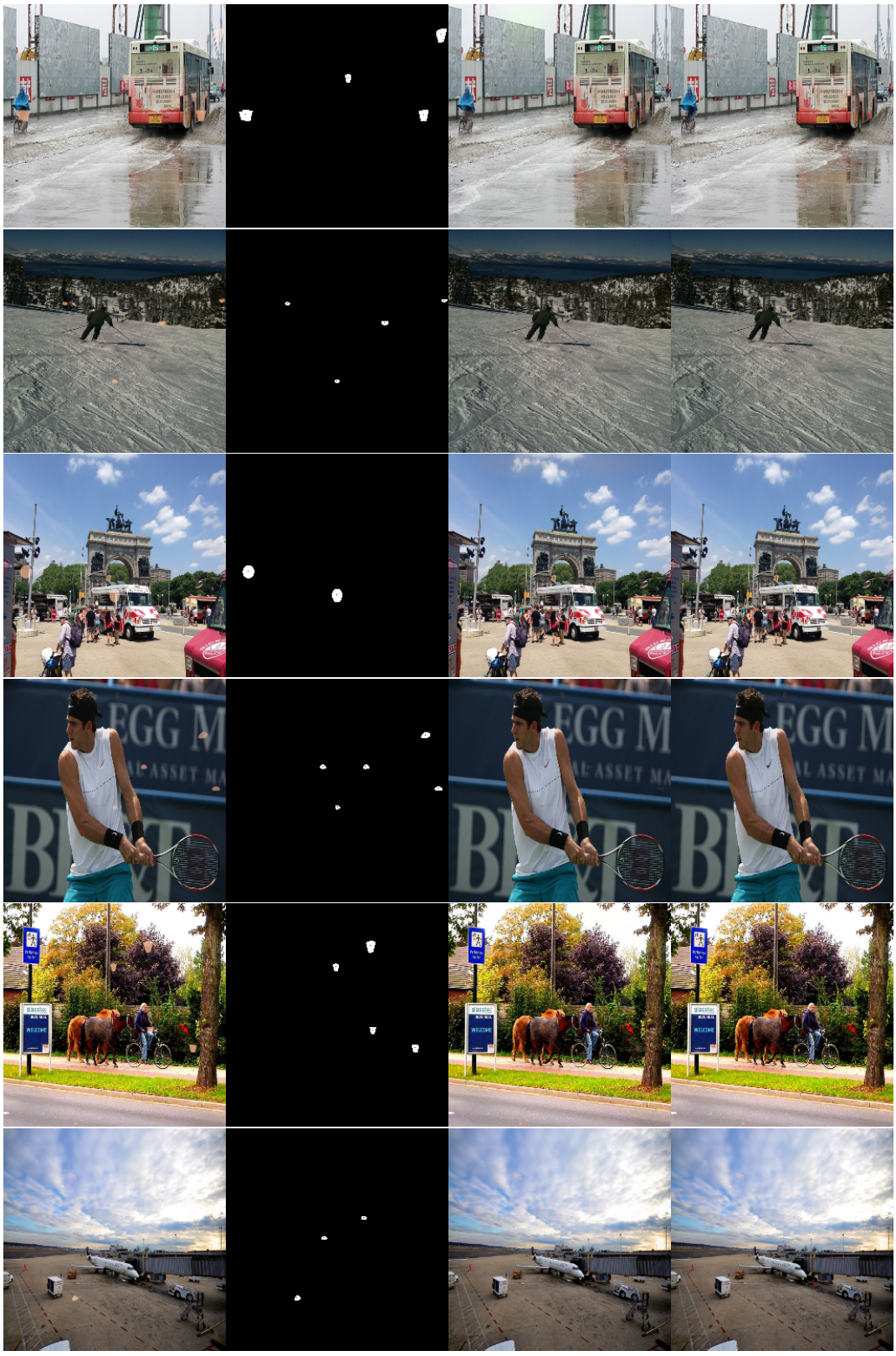

FIGURE 4.4: Results with for 4-channels input with patched loss. Setup with coefficient for the MSE and perceptual losses equal to 1, and coefficient for the adversarial loss equal to 1000 showed the most eye-pleasing results

FIGURE 4.5: Detailed view of the models output from Figure 4.4. First column - input image, second column - prediction.

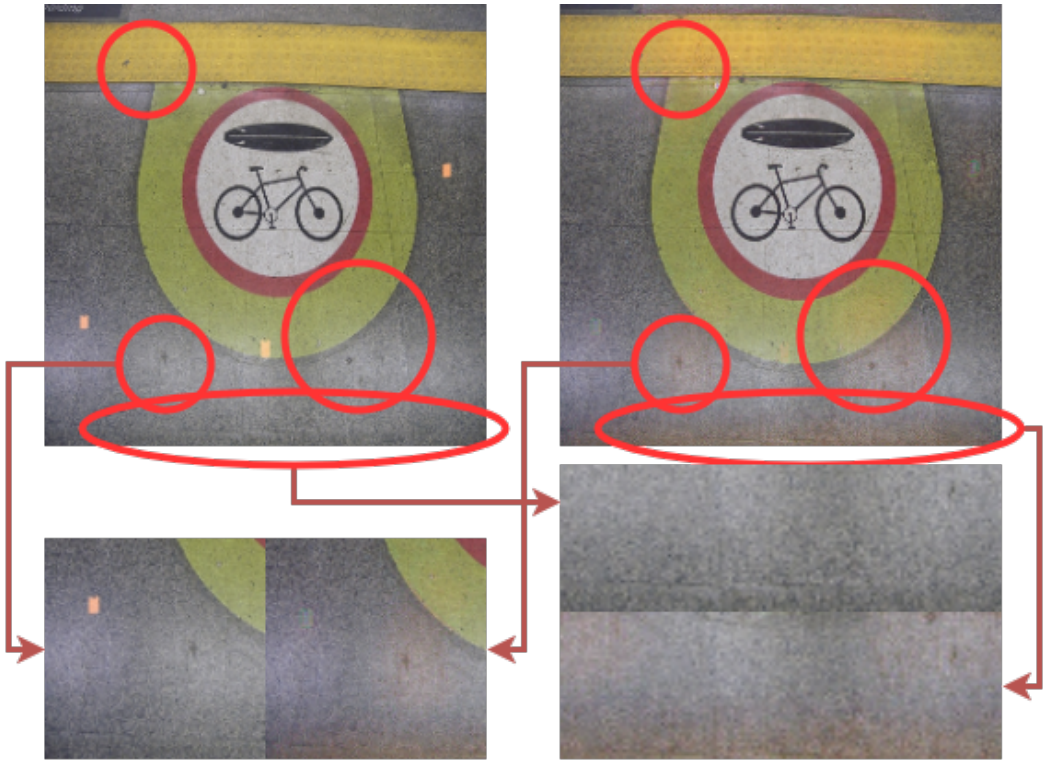

FIGURE 4.6: Artifacts caused by adding the patched loss. Likely, it is caused by the nature of the loss: we amplify penalty for parts with reflection, so the model "pays less attention" to the reflection-free part of the image

Chapter 5

Conclusion

We proposed a competitive approach for the SIRR problem, using feature pyramid and generative adversarial networks. The experiments showed that our approach generalizes better on the real-world data that the SOTA, improving SSIM by 5.6% and PSNR by 21.1%. We proposed a novel and cheaper procedure for collecting real-world data and collected samples with this approach. We addressed the problem of bright spots removal and introduced a promising approach for solving it, though many improvements and new experiments can be done further.

Bibliography

Alayrac, Jean-Baptiste et al. (2019). *Controllable Attention for Structured Layered Video Decomposition*. arXiv: 1910.11306 [cs.CV].

Chang, Yakun et al. (2020). "Siamese Dense Network for Reflection Removal with Flash and No-Flash Image Pairs". In: *International Journal of Computer Vision*, pp. 1–26.

Community, Blender Online (2018). *Blender - a 3D modelling and rendering package*. Blender Foundation. Stichting Blender Foundation, Amsterdam. URL: http://www.blender.org.

Fan, Qingnan et al. (2017). "A Generic Deep Architecture for Single Image Reflection Removal and Image Smoothing". In:

Fan, Qingnan et al. (2019). "Deep Reflection Prior". In: *arXiv preprint arXiv:1912.03623*.

Flickr (2020). URL: https://flickr.com/.

Gandelsman, Yossi, Assaf Shocher, and Michal Irani (2018). *"Double-DIP": Unsupervised Image Decomposition via Coupled Deep-Image-Priors*. arXiv: 1812.00467 [cs.CV].

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press.

Han, Byeong-Ju and Jae-Young Sim (2019). "Single Image Reflection Removal Using Non-Linearly Synthesized Glass Images and Semantic Context". In: *IEEE Access* 7, pp. 170796–170806.

He, Kaiming et al. (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Isola, Phillip et al. (2016). *Image-to-Image Translation with Conditional Adversarial Networks*. arXiv: 1611.07004 [cs.CV].

Jin, M., S. Süsstrunk, and P. Favaro (2018). "Learning to see through reflections". In: *2018 IEEE International Conference on Computational Photography (ICCP)*, pp. 1–12.

Johnson, Justin, Alexandre Alahi, and Fei-Fei Li (2016). "Perceptual Losses for Real-Time Style Transfer and Super-Resolution". In: *CoRR* abs/1603.08155. arXiv: 1603.08155. URL: http://arxiv.org/abs/1603.08155.

Kim, Soomin, Yuchi Huo, and Sung-Eui Yoon (2019). *Single Image Reflection Removal with Physically-based Rendering*. arXiv: 1904.11934 [cs.CV].

Kingma, Diederik P. and Jimmy Ba (2014). *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980 [cs.LG].

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc., pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Kupyn, Orest et al. (2019). *DeblurGAN-v2: Deblurring (Orders-of-Magnitude) Faster and Better*. arXiv: 1908.03826 [cs.CV].

Lecun, Y. et al. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324.

Li, Chao et al. (2019). "Single Image Reflection Removal through Cascaded Refinement". In: *arXiv preprint arXiv:1911.06634*.

Li, Tingtian and Daniel PK Lun (2019). "Image Reflection Removal Using the Wasserstein Generative Adversarial Network". In: *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 1–5.

Lin, Tsung-Yi et al. (2014). "Microsoft COCO: Common Objects in Context". In: *CoRR* abs/1405.0312. arXiv: 1405.0312. URL: http://arxiv.org/abs/1405.0312.

Lin, Tsung-Yi et al. (2016). "Feature Pyramid Networks for Object Detection". In: *CoRR* abs/1612.03144. arXiv: 1612.03144. URL: http://arxiv.org/abs/1612.03144.

Liu, Yu-Lun et al. (2020). "Learning to See Through Obstructions". In: *arXiv preprint arXiv:2004.01180*.

Liu, Yunfei and Feng Lu (2019). *Separate In Latent Space: Unsupervised Single Image Layer Separation*. arXiv: 1906.00734 [cs.CV].

Liu, Yunfei et al. (2019). *Semantic Guided Single Image Reflection Removal*. arXiv: 1907.11912 [cs.CV].

PyTorch. URL: https://pytorch.org/.

Simonyan, Karen and Andrew Zisserman (2014). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv: 1409.1556 [cs.CV].

Springer, Ofer and Yair Weiss (2017). *Reflection Separation Using Guided Annotation*. arXiv: 1702.05958 [cs.CV].

Sun, J. et al. (2019). "Multi-Modal Reflection Removal Using Convolutional Neural Networks". In: *IEEE Signal Processing Letters* 26.7, pp. 1011–1015.

Szegedy, Christian et al. (2015). "Going deeper with convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9.

Szegedy, Christian et al. (2017). "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning". In: URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14806.

Walt, Stéfan van der et al. (June 2014). "scikit-image: image processing in Python". In: *PeerJ* 2, e453. ISSN: 2167-8359. DOI: 10.7717/peerj.453. URL: https://doi.org/10.7717/peerj.453.

Wan, Renjie et al. (2017). "Benchmarking Single-Image Reflection Removal Algorithms". In: *International Conference on Computer Vision (ICCV)*.

Wan, Renjie et al. (2019). *Face Image Reflection Removal*. arXiv: 1903.00865 [cs.CV].

Wang, Jifeng, Xiang Li, and Jian Yang (2018). "Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1788–1797.

Wang, Zhou et al. (2004). "Image quality assessment: from error visibility to structural similarity". In: *IEEE transactions on image processing* 13.4, pp. 600–612.

Wen, Qiang et al. (2019). "Single image reflection removal beyond linearity". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3771–3779.

Yang, Yang et al. (2019). "Fast single image reflection suppression via convex optimization". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8141–8149.

Zhang, H. et al. (2019). "Fast User-Guided Single Image Reflection Removal via Edge-aware Cascaded Networks". In: *IEEE Transactions on Multimedia*, pp. 1–1.

Zhang, Xuaner, Ren Ng, and Qifeng Chen (2018). "Single Image Reflection Separation with Perceptual Losses". In: *IEEE Conference on Computer Vision and Pattern Recognition*.