CQF1

FYP Final Report

Low-Light Video Enhancement Using Deep Learning

by

Yingshan Chang, Ka Leong Cheng, Yanming Kang, Xuanyi Li

CQF1

Advised by

Prof. Qifeng Chen

Submitted in partial fulfillment

of the requirements for COMP 4981

in the

Department of Computer Science

The Hong Kong University of Science and Technology

2019 - 2020

Date of submission: April 29, 2020

Table of Contents

Abstract

Videos captured in low-light scenes are often noisy and suffer from blurs induced by either the motion of the objects or the shaking of the camera. Most of the previous work mainly focuses on motion deblurring for low-light images rather than videos. Also, there is a very limited number of public datasets regarding video enhancement for low-light scenes.

In our final year project, we explore possibilities of reducing noises and motion blurs in low-light RGB videos using deep learning methods. We propose a novel approach to collect data of noisy and blurry video frames with their corresponding bright, clear, shape images. Accordingly, we also introduce a public dataset that is collected using our collecting pipeline. We propose an end-to-end fully convolutional network pipeline with a fine-tuning strategy to help finish the work of low-light video enhancement for motion deblurring. Finally, we conduct experiments on our proposed dataset, and the experiments show that our network pipeline outperforms previous works in terms of both numerical evaluation metrics and human visual perception.

Acknowledgment

We want to thank our supervisor, Professor Qifeng Chen, for the patient guidance and support during our final year project. His words inspire us with confidence and passion in the field of low-light video enhancement. With his help throughout the project, we are short on a few detours.

We would also like to express our gratitude to our communication tutor Ms. Noorliza Daveau for her help on the report writing. Her opinions help us to construct our report clearer and more concise.

1. Introduction

1.1 Overview

One of the major application areas of video processing is to improve the pictorial information for human perception, and it has been actively researched. Most of the time, current video processing technologies achieve significant performance under normal light conditions. Nonetheless, video enhancement under low-light conditions is a much more challenging and poorly researched area.

Low-light videography is challenging not only due to the low photon counts but also the dynamic scenes in the video. Generally, there are two ways to help us make the scene visually better. One is to use traditional image processing methods based on matrix computation, which is embedded in many cameras. But this requires the user to manually adjust parameters and balance the trade-offs. For example, higher ISO makes the resulting scenes brighter while also amplifies noises. Setting the exposure time longer can also tackle the low-light condition but may bring unwanted blur due to handshake and object movement. It would be very hard for laypeople to understand the trade-offs between different parameters and find the optimal configuration.

In addition, deep-learning-based approaches have led to lots of progress in this field. However, many of these technologies require raw data as the original input, rather than the common input formats such as PNG and JPG (JPEG). Using raw data as input is feasible in the image processing field because the camera can keep raw data when taking static photos. Nonetheless, general cameras do not keep raw data when shooting videos. Hence, after splitting a video into consecutive frames, we only get RGB data, which is more compressed and contains less information. Therefore, we cannot directly use the pipeline of image enhancement to process video frames. Instead, we can study each component of current image enhancement pipelines and design a new video enhancement pipeline based on deep learning, which takes RGB data as input and is able to process a video in a frame-by-frame order.

The major contributions of this project are listed as followed:

1) Provide a new dataset with low-light videos and clear, bright images that have one-to-one correspondences with video frames.

2) Propose an end-to-end video-processing pipeline based on deep neural networks aiming at doing frame-by-frame enhancement, including denoising and motion deblurring for videos captured in low-light scenes.

1.2 Objectives

This project aims at training a deep learning model on a manually collected, small-scale dataset, which can convert a blurred low-light video to a high-quality video with enough brightness and less blur. Our model achieves the following objectives:

- **Bright & clear:** This is the primary purpose of our low-light video enhancement problem. Brightness and clarity are directly correlated with the visual quality of a video. They are also the two main indicators we use later in the perceptual evaluation.
- **No artifact:** It is not a rare issue that the output of some computer vision models contains artifacts, such as color distortion or dark spots. Thus, it calls for special attention. Even if these artifacts do not greatly affect numerical evaluation scores, it is unacceptable that the visually unpleasant artifacts exist in the model output.
- **Robustness:** Our model should be able to deal with a variety of objects in a scene, including text, fine patterns, background, shadow, and so on. High robustness indicates strong generalization property and wide application field.
- **Good performance on different evaluation metrics:** Our method should demonstrate originality and surpass the existing models in certain aspects.

The project is divided into the following steps to achieve the objectives:

- **Collect a dataset:** Deep learning requires a large amount of data with inputs and corresponding output to supervise the parameter optimization process. As there is no existing real dataset for low-light video enhancement and we regard synthetic data not effective, collecting the dataset on our own is essential.

- **Model design, implementation, and training:** With the dataset ready, we need to come up with a new algorithm and train a deep learning model that can produce clear and bright output given the low-light blurry videos. Our ground-truth images will participate in the loss calculation and assist the model training during the gradient descent process.

- **Testing, evaluation, and analysis:** The last stage of this project is to compare our results with other advanced research, analyze the gains and losses, and demonstrate the contribution of our model.

1.3 Literature Survey

Perceptual quality enhancement of static images in low-light scenes has been extensively studied in the literature. Quite a few studies have also investigated motion deblurring in consecutive video frames. In this section, we give a short review of existing approaches for both image and video enhancement. Then we briefly discuss which existing techniques are applicable to our problem.

1.3.1 Data Collection

Obtaining pairs of the corresponding low-light, blurry videos and sharp, bright videos for training is a big challenge. We widely study the existing methods in the literature and conclude that they are not suitable for our low-light video problem. Finally, we adopt a method that is less used in the literature but can better approximate the data in the real world.

1.3.1.1 Synthetic Data

A typical approach in the literature is to use a high frame-rate camera to capture only the sharp, bright static images, then synthesize the blurred ones by convolving the images with motion-blur kernels. This method is widely used in image deblurring or denoising network, and several research teams [1,2] can simulate the motion blur in the image taken in the real world. However, after some feasibility assessment, we find that this approach is not applicable to our problem. We require that the synthetic video frames should have consistency so that when they are concatenated together, it should become a complete video. Such consistency is hard to achieve, not only in the synthetic motion blur but also in the synthetic “low-light”. Another “semi-synthetic” approach

is to use the average of sharp frames from videos [1,2,3]. But in order to study video deblurring in low-light conditions, we still have to synthesize the low-light effect, which also introduces a domain gap where it is possible that the model is trained on synthetic data and fails to generalize to real-world data.

1.3.1.2 Real-World Data

As suggested by [4], the video pair can be taken by shooting the same moving scene twice with long and short exposure time. The long exposure time will make the resulting scenes brighter, while the short exposure time results in dark scenes. However, scenes captured by the long exposure time are also sensitive to handshaking or object motions. Thus, only static images can be taken using a long exposure time.

1.3.2 Deep Network Architecture

1.3.2.1 Convolutional Neural Network (CNN)

CNNs has dramatically improved the baseline in many tasks about computer vision. A convolutional net is usually built on the basic components, including convolutional-layer, batch-normalization-layer, activation-layer, and pooling-layer. In the literature [5], the convolutional, pooling, batch-norm, and activation layers are usually combined into one convolutional block. The whole net will be multiple convolutional blocks with different hyperparameters stacked together. It learns an end-to-end mapping between a low-light input and a high-quality output by learning a nonlinear mapping between every two neighboring layers.

1.3.2.2 U-Net Structure

U-Net is shaped like an hourglass, with down-sampling layers to reduce the input to a low-resolution feature map in the first half, as well as up-sampling layers to restore the feature map to the original resolution. The feature map in the last layers is considered as the output. U-Net is originally proposed for image segmentation but is proved to be effective for other tasks where the input and output have the same size [5,6]. Besides, U-Net has also been shown to achieve efficient GPU consumption [5].

1.3.2.3 Skip Connections

Skip connection is a widely used trick in deep learning. It connects non-adjacent layers by a “shortcut link”, allowing a more direct flow of local information from shallower

layers to deeper layers. [7] This prevents the gradual loss of information in the propagation process through the deep network, so as to achieve better convergence. Regarding a combination of U-Net structure and skip connections, it is a good way to connect hidden layers with identical size from the down-sampling path to the up-sampling path. [5]

1.3.2.4 Generative Adversarial Network (GAN)

GAN [8] is a powerful generative model that casts the generative task as a game between two networks: a generator network producing synthetic data and a discriminator network trying to distinguish synthetic data from real data. The purpose of the generator network is to mimic the distribution of real data so that it will give the discriminator a hard time to classify the generated data and the real data. [9] introduces conditional GAN (cGAN) to an image deblurring task and achieves better perceptual quality. Inspired by this, we seek to find out whether the conditional GAN structure will be effective for low-light image enhancement as well.

1.3.3 Loss Function Design

The success of deep neural networks largely depends on the design of loss function, as it determines the target of parameter optimization. The loss function should have a mathematical meaning consistent with our subjective expectation; otherwise, the training process will easily deviate from what is expected.

1.3.3.1 Pixel-Wise Loss

Pixel-wise loss is the most common loss function. It measures the difference between two images by calculating the pixel-wise absolute error or mean square error. However, it is rarely used as a standalone loss function [10] as the network that is trained with the MSE loss tends to find the average of plausible solutions, without selective emphasis on different objects or textures in an image. This usually leads to an “over-smoothing” effect and a lack of high-frequency features such as edges and corners. [7] Hence, pixel-wise loss needs to be paired with other loss functions in order to minimize the gap between the assessing metrics and the human visual system.

1.3.3.2 Perceptual Loss

Some advanced approaches such as [6,10] estimate the perceptual similarity by mapping both the model output and ground truth into a higher-dimensional feature

space (e.g., a pre-trained VGG network), then compare the MSE between two feature maps. [10] also investigates the nature of CNN feature space in different layers. In a nutshell, in shallow convolution layers, each neuron has a small receptive field. So, these layers extract low-level spatial information that mostly represents edges and blobs. In deep convolution layers, neurons have a large receptive field and tend to learn semantic features with global meanings, such as textures and abstract objects. In that sense, by selecting multiple feature spaces from shallow to deep and calculate the perceptual loss in a parallel, we can achieve the goal of considering both fine-grained details and global features.

1.3.3.3 Wasserstein Loss

Wasserstein loss is proposed by Arjovsky in [11]. It points out that the loss function in GAN measures the JS divergence, which is a poorly designed metric for the similarity between two distributions. [11] proposes a new loss function based on the Wasserstein distance and concludes that it has better theoretical properties. in other words, WGAN can avoid mode collapse and is conducive to faster convergence [9].

1.3.3.4 Contextual Loss & Contextual Bilateral Loss

The contextual loss [12] is proposed to handle unaligned pairs of training data. It treats the input image as a set of features, which are usually computed by a pretrained network. Similarly, the target image is also treated as a set of features, and each input feature is matched to its nearest neighboring feature in the target image. The distance metrics can be L_1, L_2, or cosine distance. The contextual loss is the average distance over all features in the image pair. Contextual bilateral loss [13] is an improved version of contextual loss with additional consideration of information in the spatial domain.

2. Methodology

2.1 Design

The design of our project includes the following aspects, namely, data collection, data preprocessing, neural network architecture, and loss function. In this part, we specify our expectations and concerns for each aspect, and we propose our solution in the next section.

2.1.1 Data Collection

As mentioned in the Objective section, there is no existing dataset for our direct usage. So, the first thing to do is to manually collect a new dataset. We use a high-quality digital SLR camera that can record videos and take clear pictures for data collection.

- For the input, we want some relatively low-quality videos (i.e., dark and blurry), we decide to use the video recording mode to collect data, as the frame quality of a video is much lower than the quality of a static picture taken by the same camera, in terms of resolution and blurry level.

- For the ground truth, we want relatively high-quality videos without blur (i.e., bright and clear), so we decide to collect a sequence of images, with one-to-one correspondences to certain video frames (i.e., they have identical scene), using the picturing mode.

The biggest challenge we expect to face in the data collection process is to ensure the dynamic alignment of the low-light input videos and the ground truth image sequences. In our proposed solution, we manage to capture the same scene twice with under different camera settings and make sure that approximately 20% of frames in the input video has its counterparts (i.e., sharp, bright images) in the ground truth.

2.1.2 Data Preprocessing

Data preprocessing is particularly important for deep learning, as we need to remove noise, delete abnormal data, and convert them into a format acceptable to the neural network. Here are several points that call for more attention：

- **Check the video qualities:** Drop the input videos that are too dark as well as the ground truth images with unwanted blur.

- **Video frame alignment:** Split videos into video frames and find the one-to-one mapping between the ground truth image and video frame. Alignment accuracy will directly affect deep learning performance.

2.1.3 Design the Neural Network Architecture

We develop the basic idea of the general structure of our neural network based on several existing methods. Our model needs to support two functions: **brightening** and **deblurring**. As there are very few existing methods that implement both functions, we separately explore models that support deblurring and brightening functions in the literature. After summarizing their characteristics, we design a general framework that can well integrate these characteristics.

- **Brightening function:** The realization of this function is based on multi-layer convolution blocks, as mentioned in *Learning to See in the Dark* [4] proposed by *Chen et al.*, and each convolution block contains a convolution layer, an activation layer, a batch normalization layer and a pooling layer. Following the design philosophy of U-Net, the number of convolutional-filters increases first and then decreases as the network goes deeper and deeper.

- **Deblurring function:** Inspired by *Motion Deblurring in the Wild* [14] *by*, we include two main components in the network, which play a significant role in generating sharp images. Firstly, skip connection connects the down-sampling and up-sampling layer with the same scale. The idea behind this is to reduce the complexity of reconstruction by making the network only generate a residual image, which is added to the blurry image in the down-sampling layer. Secondly, the U-Net structure with decreasing and increasing scales enables the network to breakdown the blur restoration task into different scales and generate a sharp image from a compressed scale gradually up to the original scale.

2.1.4 Design the Loss Function

A loss function in deep learning is a function that usually represents the “cost” that mathematically represents the discrepancy between the model output and our subjective expectation. During model training, the parameters are adjusted along the direction

of minimizing the loss function, so that the model output will move closer to what we intuitively want. We have studied several loss functions that have been experimentally proved helpful in this field. We try pixel-wise loss in the preliminary training process. Then we investigate other losses to guide the reconstruction of perceptual quality. Eventually, we incorporate multiple loss components with different weights to obtain the overall loss.

2.2 Implementation

2.2.1 Data Collection

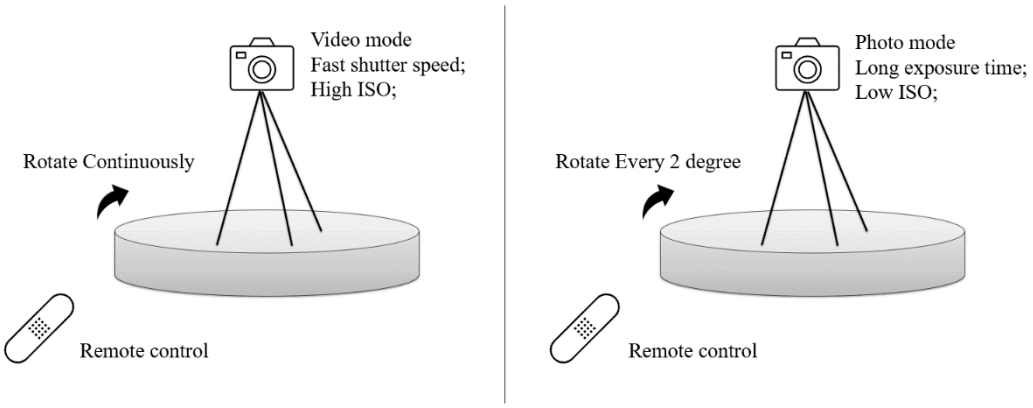

Fig. 2.1. Illustration of our data collection method. The left-hand side shows the way to collect blurry, noisy dark videos as input; The right-hand side shows the way to collect clearer brighter frames as ground truths.

- **Dynamic alignment:** We solve this problem with the concept of relative motion in physical, letting the camera move according to the same path so as to capture the same moving scene. As shown in Fig. 2.1, what we need are two things: a digital SLR camera and a turntable. We fix our digital SLR camera onto the turntable. With the movement of the turntable, we can mimic the general moving scene, and all the devices are controlled by remote control, making sure no human action is involved during the video taking process.

- **Scene setting:** We choose indoor environments with a weak light source as the scene for data collection. The scene contains only limited natural lights in the evening coming from the window.

- **Input:** To get the videos that are dark and blurred, we set the shutter speed at 1/30, ISO 2000, and aperture size f/3.5. Every input video lasts for about 3 seconds on average, with a frame rate of 24 fps.

- **Ground truth:** We use a rotary table that can uniformly rotate by 2 degrees (4×0.5 degree) every time, which is controlled by remote control to make sure that the moving tracks can be identical for both times of shooting. Meanwhile, we keep everything around to be the same during the two corresponding video shooting. When taking image sequences for ground truth, it is very hard for us to take clear and bright in a dark scene. Thus, we use the long-exposure time, which is set automatically by the camera Nikon D5500 to shoot multiple frames statically by moving slightly along the moving tracks for each frame. We combine the images together to have the image sequences for ground truth. Every of the image sequences we collect for ground truth contains exactly 24 frames to match the frame rate of input videos.

  The data we collect contains a variety of scenes. As shown in Fig. 2.2, in each scene, we collect a pair of a low-quality, dark and blurry video and a sequence of clear and bright pictures. For each pair video, their contents and motion are the same, and their lighting conditions are different.

Fig. 2.2. Sample Data. A is a low-quality dark blurry video. B is a sequence of clear and bright images.

2.2.2 Data Preprocessing (MATLAB)

- **Step 1: turn the input videos into frames set**

  The video inputs might add complexities for training, so we cut the video into image frames according to the frame rate when we shoot the video (24 fps). Our 3s videos become 72 images each for the frame rate of 24 fps.

- **Step 2: data selection (filtering)**

  According to experience, it is hard to train the model with too dark input frames or unclear ground truths. Therefore, it is important to filter out samples with too dark inputs and too blurred ground truths. We filter out the samples with an average intensity of the video frames, which is less than 20. For the blurred ground truth, we do the filtering manually with our eyes. In Fig. 2.3, we demonstrate some selection criteria.

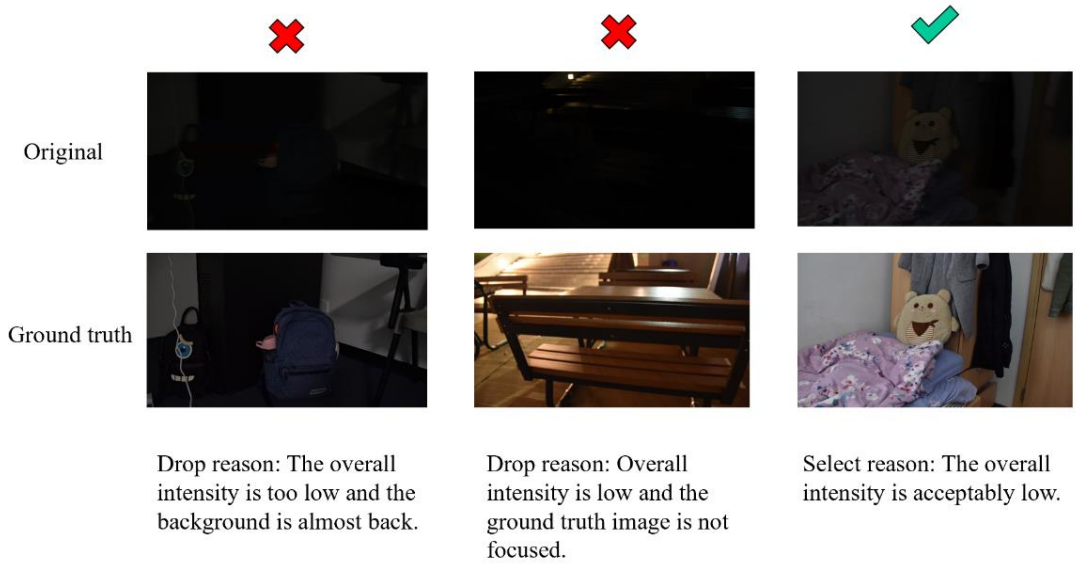

Fig. 2.3. This illustrates the types of data that are not suitable for our model and would be dropped.

- **Step 3: find input & ground truth pairs**

  After getting the corresponding pairs of input and ground truth frames set, we need to find the corresponding input frame for each ground truth frame in the image sequences. We manage to find such a pair though the build-in function of the 2-D correlation coefficient in MATLAB. Specifically speaking, we first read in all the input frames and ground truth frames in gray scale and do the histogram equalization. Then, for every ground truth frame, we compare it with all the input frames and select the input frame with the highest score as its corresponding inputs. Therefore, for every pair of 3s video and 24 image sequences, we are able to collect 24 pairs of input and ground truth frames. The simple diagram for the whole process is shown in Fig. 2.4

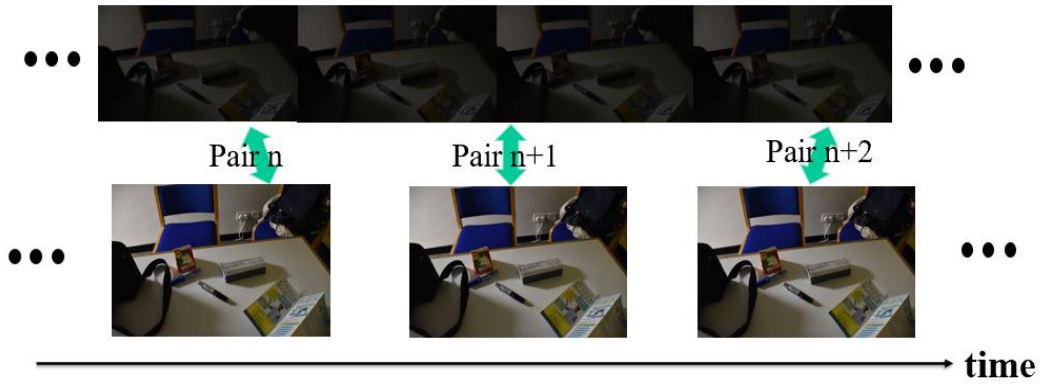

Fig. 2.4. Find input and ground truth data pair in dark videos and clear images.

- **Step 4: image registration (translation)**

  With pairs of input and ground truth frames, although they all have the relative highest score, still every pair does not have a perfect match due to some horizontal and vertical offset, so we need to image registration for every pair of frames. A remark for this step is that we do not use rigid transformation (i.e., rotation factors) for image registration, because it turns out that translation gives a better result for rigid registration.

- **Step 5: cut frames into small frames**

  Up to this step, the size of each frame is 1920×1080, which is too large for deep learning training, due to the limitation of GPU size. Therefore, in the final step, we cut the image into small pieces of size 512×512. In other words, we will get six pairs of frames of size 512×512 for each pair of original frames. We show some sample pairs of frames in Fig. 2.5, where the images in the first row are input dark and blurred frames, and their corresponding ground truth images are in the second row.

Fig. 2.5. Sample data pairs of size 512×512.

2.2.3 Model Architecture

Formally, our model architecture aims to recover a bright and clear target frame Y, given only an input dark and blurry frame X. No information about the blur kernel is provided, as we are using real data instead of synthetic data. The recovery is done by a conditional GAN network, consisting of a generator and a critic (discriminator), which are trained in an adversarial manner with Wasserstein Generative Adversarial Network - Gradient Penalty (WGAN-GP) loss [15]. On top of that, we also utilize the fine-tune strategy with Contextual Bilateral (*CoBi*) loss [13] to fine-tune our model for sharper and more structural output images. An overview of our network is shown in Fig. 2.6.

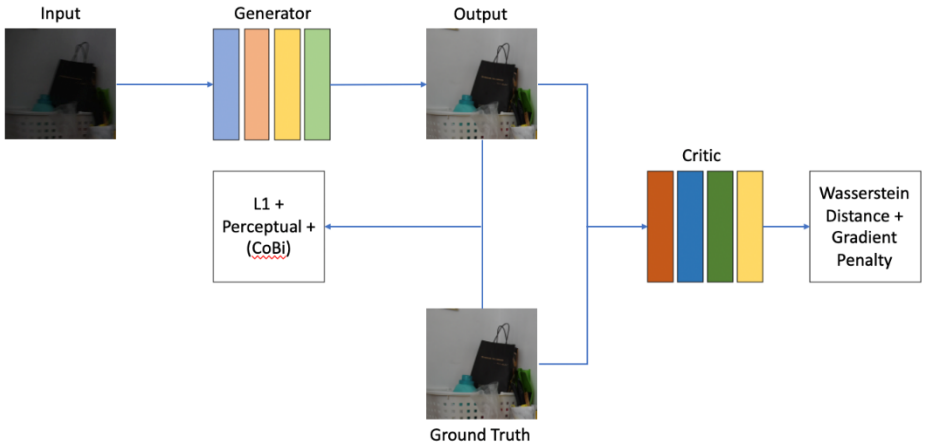

Fig. 2.6. Overview of our network. The network is designed using GAN architecture with a generator and a critic to compete with an adversarial loss. *CoBi* loss is applied to the generator during fine-tuning.

2.2.3.1 Generator

The GAN architecture consists of a generator module, denoted as G, aiming to generate images that have little difference with the corresponding ground truth images.

2.2.3.1.1 Generator Architecture

For the generator, we use the basic U-Net architecture, which is an encoder-decoder with skip connections between mirrored layers in the encoder and decoder stacks. The encoder will down-sample the input image to a small feature map using convolutional layers with a stride greater than 1, and the decoder will up-sample the feature map using bilinear interpolation. One significant step in the decoder is adding the corresponding convolutional result to the current deconvolution result. By adding the mirrored result, the output can successfully maintain different levels of features of the observed input images. The ideas of encoder-decoder and U-Net are shown in Fig. 2.7.

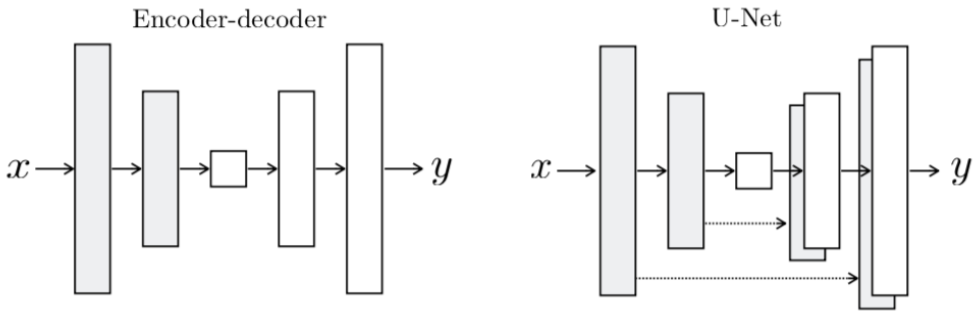

Fig. 2.7. Encoder-decoder architecture and U-Net architecture.

U-Net is based on multi-layer convolution blocks, and each convolution block contains a convolution layer, a batch normalization layer, an activation layer (LeakyReLU), and a max-pooling layer. Additionally, we employ the idea of ResNeXt [16] Blocks into the U-Net structure, which helps to reduce the risk of over-adapting the hyperparameters to a specific dataset. Compared with the normal Res Block, ResNeXt Block introduces an additional concept of “cardinality”. Cardinality is the path in-between the convolution, and the dimension of cardinality controls the number of more complex transformations. The general idea of ResNeXt block is explained in Fig. 2.8.

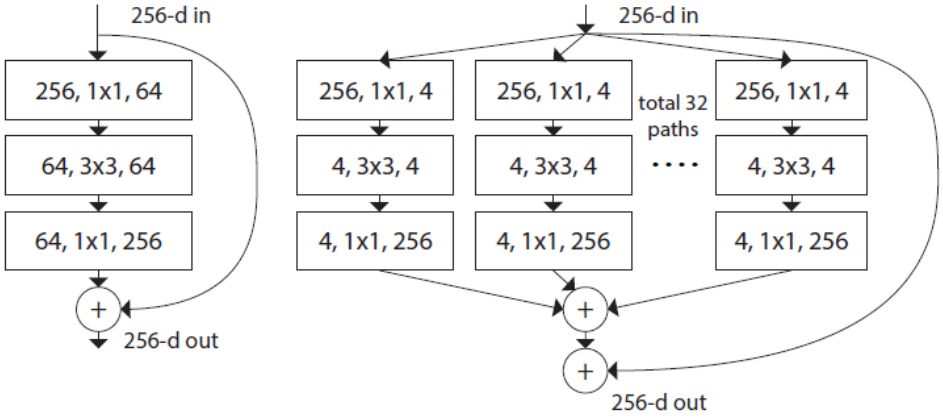

Fig. 2.8. Residual Block in ResNet (Left), A Block of ResNeXt with Cardinality = 32 (Right).

Formally, each batch of input images is a tensor with shape (N, C, H, W), where N denotes the batch size, C denotes the number of channels, and H, W denotes the height and width of the frames. Then the process of generator G can be described as:

$$y_{restored} = G(x), \qquad \text{Eq. 2.1}$$

where the output is of shape (N, C, H, W) as well.

2.2.3.1.2 Content Loss

Pixel-wise loss tends to yield "over smoothed" output. Therefore, using a standalone pixel-wise loss is not optimal. So, we measure the content loss of the generator, using an equal-weighted the pixel-wise L_1 loss and perceptual loss.

***L_1* Loss.** It represents the sum of all the pixel-wise absolute differences between the ground truth the model output. We do not use L_2 loss, because compared with L_1 loss, though it is smooth at zero points but is more sensitive to noises or outliers. The formula of L_1 loss is expressed as:

$$L_1 = \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W}|y_{true}(i,j) - y_{restored}(i,j)|. \qquad \text{Eq. 2.2}$$

Perceptual Loss. We calculate perceptual loss works by averaging all the pixel-wise errors between two features maps generated from a pre-trained VGG-16 net. We chose VGG-16 as the feature extraction network and extracted three feature maps in shallow (Conv1-2, Conv2-2), middle (Conv3-3), and deep (Conv4-3) layers respectively to increase feature complexity and receptive field, as shown in Fig. 2.9.

The idea behind this is to comprehensively compare the similarity between two images in terms of both low-level spatial details and global visual effects.

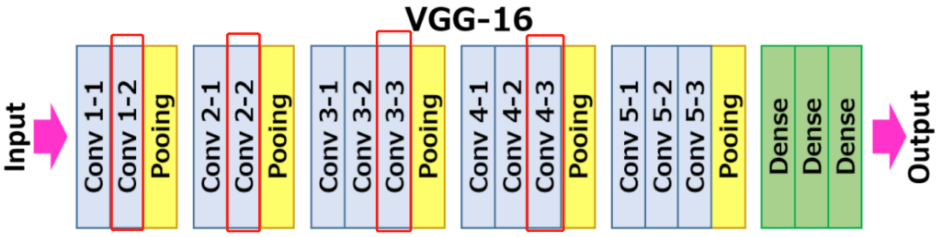

Fig. 2.9. Perceptual loss based on pre-trained VGG-16.

We can then formulate the perceptual loss as followed:

$$L_p = \frac{1}{H_{i,j}W_{i,j}} \sum_{x=1}^{H_{i,j}} \sum_{y=1}^{W_{i,j}} ||\phi_{i,j}(y_{true})_{x,y} - \phi_{i,j}(y_{restored})_{x,y}||_2. \quad \text{Eq. 2.3}$$

In the equation, $\phi_{i,j}$ is the feature map obtained by the j[th] convolutional layer and before the i[th] max-pooling layer within the VGG16 network, pretrained on ImageNet [17]. $W_{i,j}$ and $H_{i,j}$ are the dimensions of the feature maps.

2.2.3.2 Critic (Discriminator)

The conditional GAN (cGAN) architecture also consists of a critic (discriminator) module, denoted as D. cGAN is to define a game between two networks, namely the generator module G, and the critic module D. The generator tries to generate a fake image based on the input image. A normal discriminator aims to distinguish between the generated images and the corresponding ground truth image. Differently, with WGAN-GP loss, the critic network D takes either a restored image or a true image as input and outputs a critic score map of size (H', W'), as shown in Eq. 2.4, where H' and W' are the height and width of the feature map.

$$Score_{critic} = \mathrm{D}(y). \quad \text{Eq. 2.4}$$

2.2.3.2.1 Critic Loss

Wasserstein Generative Adversarial Network (WGAN) has achieved good results on both style transfer and image super-resolution, and we find that their loss function works well as our critic loss on our task. Usually, GANs are hard to converge during training. A variation of WGAN loss with gradient penalty [15] helps solve this problem

by calculating a leveraged Wasserstein distance of joint distributions to stabilize training. Due to the limited GPU resource, the training stabilization is crucial for low-light image deblurring, because it allows to use a relatively lighter neural network architecture to replace the normal choice of very deep neural networks, significantly increases the robustness of generator architecture choice. The Wasserstein distance, or namely the critic loss, is the sum of the average critic score on real images and the negative sum of the average critic score on restored images. The formulation of the critic loss is as followed:

$$L_{critic} = \frac{1}{H'W'} \sum_{i=1}^{H'} \sum_{j=1}^{W'} D(y_{true})_{i,j} - \frac{1}{H'W'} \sum_{i=1}^{H'} \sum_{j=1}^{W'} D(y_{restored})_{i,j}. \quad \text{Eq. 2.5}$$

2.2.3.3 Fine Tuning

We also introduce a fine-tuning stage for the generator, by making use of an additional Contextual Bilateral (*CoBi*) loss into the content loss.

Contextual Loss. It is designed to handle unaligned pairs of training data. It treats both the input and the target images as sets of features (for our implementation, they are extracted from a pretrained VGG-19 network). Each feature of the input image is matched to its nearest neighboring feature in the target image. The distance metrics can be L_1, L_2, or cosine distance. Among them, cosine distance gives the best output both visually and numerically. The Contextual loss averages the distances over all features in the image pair. As showed in Eq. 2.6, $CX(P, Q)$ is the contextual loss between two feature maps *P* and *Q*, where $\mathbb{D}_{p_i,q_j}$ means the distance between p_i and q_j.

$$CX(P, Q) = \frac{1}{N} \sum_{i}^{N} \min_{j=1,\dots,M} \left(\mathbb{D}_{p_i,q_j} \right). \quad \text{Eq. 2.6}$$

Contextual Bilateral Loss. *CoBi* loss is an improved version of the contextual loss. Contextual loss sometimes creates strong artifacts because contextual loss does not consider the spatial structure. For *CoBi* loss, it considers local contextual similarities with weighted spatial awareness. The formula of *CoBi* loss is defined as followed:

$$CoBi(P, Q) = \frac{1}{N} \sum_{i}^{N} \min_{j=1,\dots,M} \left(\mathbb{D}_{p_i,q_j} + w_s \mathbb{D}'_{p_i,q_j} \right), \quad \text{Eq. 2.7}$$

where $\mathbb{D}'_{p_i,q_j} = ||(x_i, y_i) - (x_j, y_j)||_2$, (x_i, y_i) and (x_j, y_j) are the spatial coordinates of features p_i and q_j, and w_s represents the weight of spatial awareness when searching the nearest neighbor features.

2.2.3.4 Loss Function

Apart from the generator's content loss and the critic loss, adding regularization to prevent overfitting is a common practice. The total variation regularization is usually used in image denoising. It measures the total difference between the adjacent pixels. At the same time, L2 regularizer on parameters is added for regularization as well.

In summary, our formulated total loss function is defined as followed:

$$L_{total} = L_{content} + \lambda_1 \cdot L_{critic} + \lambda_2 \cdot L_{reg}, \quad \text{Eq. 2.8}$$

where λ_1 and λ_2 are the weight factors.

2.3 Testing

We conducted experiments on our collected dataset. We detail the experimental setup, the results, and the ablation studies in this section.

2.3.1 Experimental Setup

2.3.1.1 Dataset and Running Environment

The dataset is self-collected and pre-processed by us, using the pipeline mentioned in previous sections. We choose indoor environments with weak light sources as the scene for data collection. The scene contains only limited natural lights in the evening coming from the window.

All inputs are dark and blurry RGB images of size 512×512×3, and all the corresponding targets are bright and clear RGB images of the same size of 512×512×3. There are in total of 1,278 image pairs in our dataset. We split the dataset into 80% of training data and 20% of testing data. Inside the training data, we will use 20% of it for validation and parameter tuning purposes.

Our model is implemented in Python3 with PyTorch support and tested with the help of the GPU resources on Co-Lab provided by Google.

2.3.1.2 Training Details

The network setting for the generator and critic used in our experiments are shown in Fig. 2.10, Fig. 2.11, respectively. Batch normalization is added after every convolutional layer to accelerate training. We use Adam optimizer for training and set the initial learning to 0.0002, and the weight factor λ_1 and λ_2 are empirically set to be 10^{-4} and 0.1. We train our network for 30 epochs and reduce the learning rate by half after epoch 10, 20, 25. We further fine-tune our model for 20 epochs with *CoBi* loss and reduce the learning rate by half again after epoch 10 during the fine-tuning process.

We first normalize all the pixel values of the input and target images within the range of [0, 1]. To prevent our network from overfitting, we apply some data augmentation techniques for the input images during training. For geometric transformation, input images are randomly flipped horizontally and vertically, rotate by a certain degree. For color channels, the RGB channels of the input images are randomly swapped. Additionally, to make our network more robust to different motion noise, gaussian noise with mean 0 is added to the input images during training. The standard deviation of the gaussian noise is also sampled from another gaussian distribution with mean 0 and standard deviation of 2/255. By doing so, different strength of noise can be added to the input images.

Finally, we clip those pixel values of the augmented input images that are outside the range of [0, 1] to ensure the final range of the input images are within [0, 1].

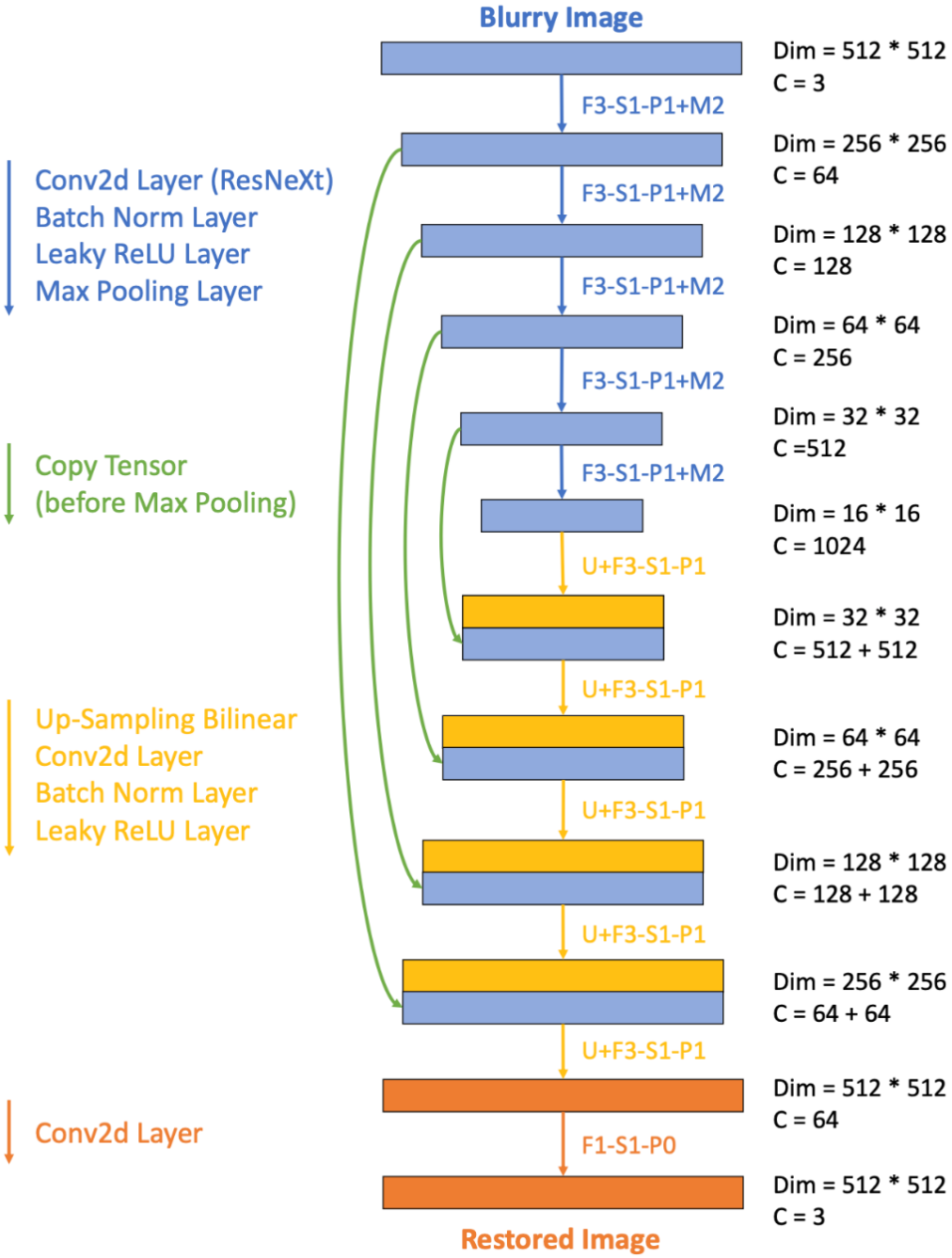

Fig. 2.10. The network setting for our generator. F, M refer to the convolution filter size and the max-pooling filter size, respectively. S, P refer to the stride and padding size of convolution, respectively. Numbers aside are their actual size.

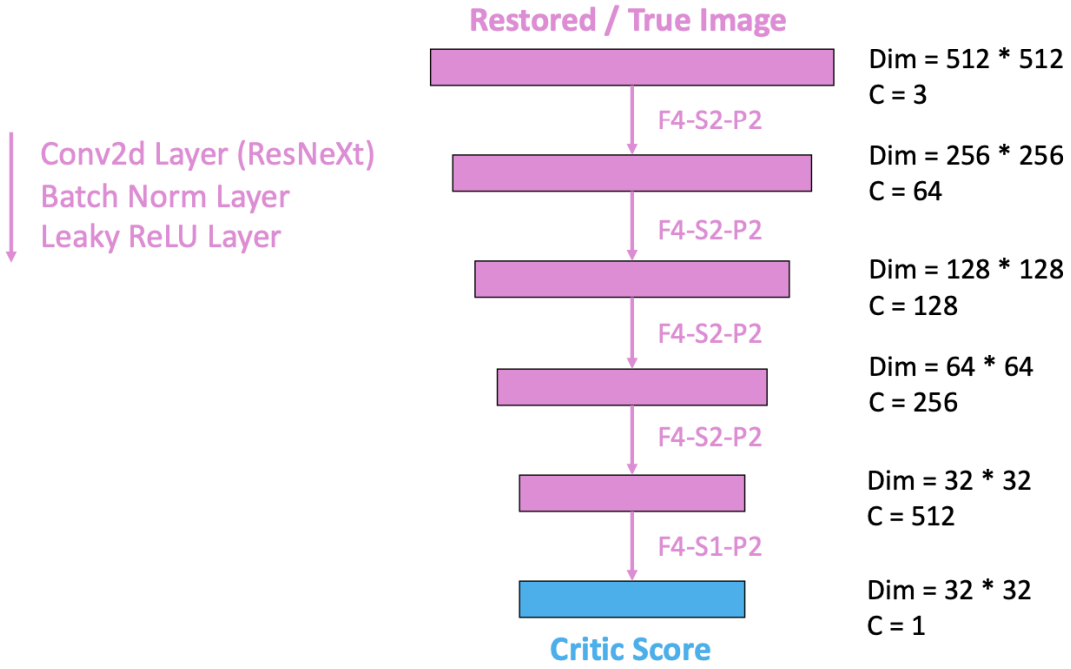

Fig. 2.11. The network setting for our critic. F, M refer to the convolution filter size and the max-pooling filter size, respectively. S, P refer to the stride and padding size of convolution, respectively. Numbers aside are their actual size.

2.3.1.3 Evaluation Metrics

To quantitatively evaluate the effectiveness of our model, we choose the two standard numeric measures that are used to determine the quality of the restored images.

Peak Signal to Noise Ratio (PSNR) measures the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation and is commonly used in image quality assessment. The formulation of PSNR is shown as followed:

$$PSNR = 20\ log_{10}\left(\frac{MAX_f}{\sqrt{MSE}}\right). \quad \text{Eq. 2.9}$$

Structural Similarity Index (SSIM) measures the calculated on various windows of an image. The formulation for measuring between two windows x and y of common size N× N is showed in Eq. 2.8, where μ_x, μ_y are the average of x and y, σ_x, σ_y are the variance of x and y, and σ_{xy} is the covariance of x and y.

$$\boldsymbol{SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}}. \quad \text{Eq. 2.10}$$

Apart from the quantitative evaluation, human perceptual evaluation can also be used for comparing our method on dynamic video deblurring with other approaches. To determine whether the converted videos are bright and clear, the human perceptual is absolutely a very reliable and important criterion. We evaluated the output videos from their brightness, clearness, color distortion, etc.

2.3.2 Results

We give a thorough comparison between our network and some previous on our dataset. The results on the testing set are shown in Table 2.1. The results of previous methods are reported after fine-tuning on our dataset for 20 epochs.

	PSNR (val)	**SSIM (val)**	**PSNR (test)**	**SSIM (test)**
MBLLEN w/o pp[1] [18]	20.33	0.627	20.30	0.616
MBLLEN w/ pp [18]	12.05	0.549	10.89	0.540
DeblurGAN [2]	22.33	0.651	23.03	0.633
Ours	**23.73**	**0.681**	**23.56**	**0.679**

Table 2.1. Numerical performance comparison on our validation set and testing set.

We can see that our network achieves the most state-of-the-art performance for our dataset in terms of both PSNR and SSIM on the testing set, which are 23.56 for PSNR and 0.679 for SSIM. They are much higher than those of previous works.

To further demonstrate the effectiveness of our model, some representative results are visually shown in Fig. 2.12 and Fig. 2.13. The result images from MBLLEN generally have color distortion. Result images before post-processing have too low contrast, while those after post-processing have too high contrast. MBLLEN is full convolutional without GANs; a sole CNN Architecture tends to generate images with unrealistic contrast, maybe the contrast is generally based on the "average" contrast among all the training images. The result images from DeblurGAN also has the problem of color distortion. Although DeblurGAN generates images with relatively high PSNR and SSIM, the generated visual images are much blurrier than all other methods, even

[1] Here 'pp' stands for post process.

blurrier than the input images; we can claim that DeblurGAN fails to handle with the motion blur in low-light scenes. Maybe it is because of the absence of *CoBi* Loss enhancement, which takes into account the spatial information. Our method achieves relatively good visual effects. Our generated images have natural colors, with good contrast and brightness, and most importantly, our method can deblur the input images to some extends. However, some minor noise and blur still exist.

In Fig. 2.14, there are some failed cases. One failure in our model is that the deblur functionality is not as expected. For motion blur, it still exists after our enhancement. Another failure case is when there is a strong light in a dark scene. This creates a large artifact in our model. The pixels around the light source tend to be black, as our model could not resolve their values. Although, in other models, the artifact also exists, it is less obvious.

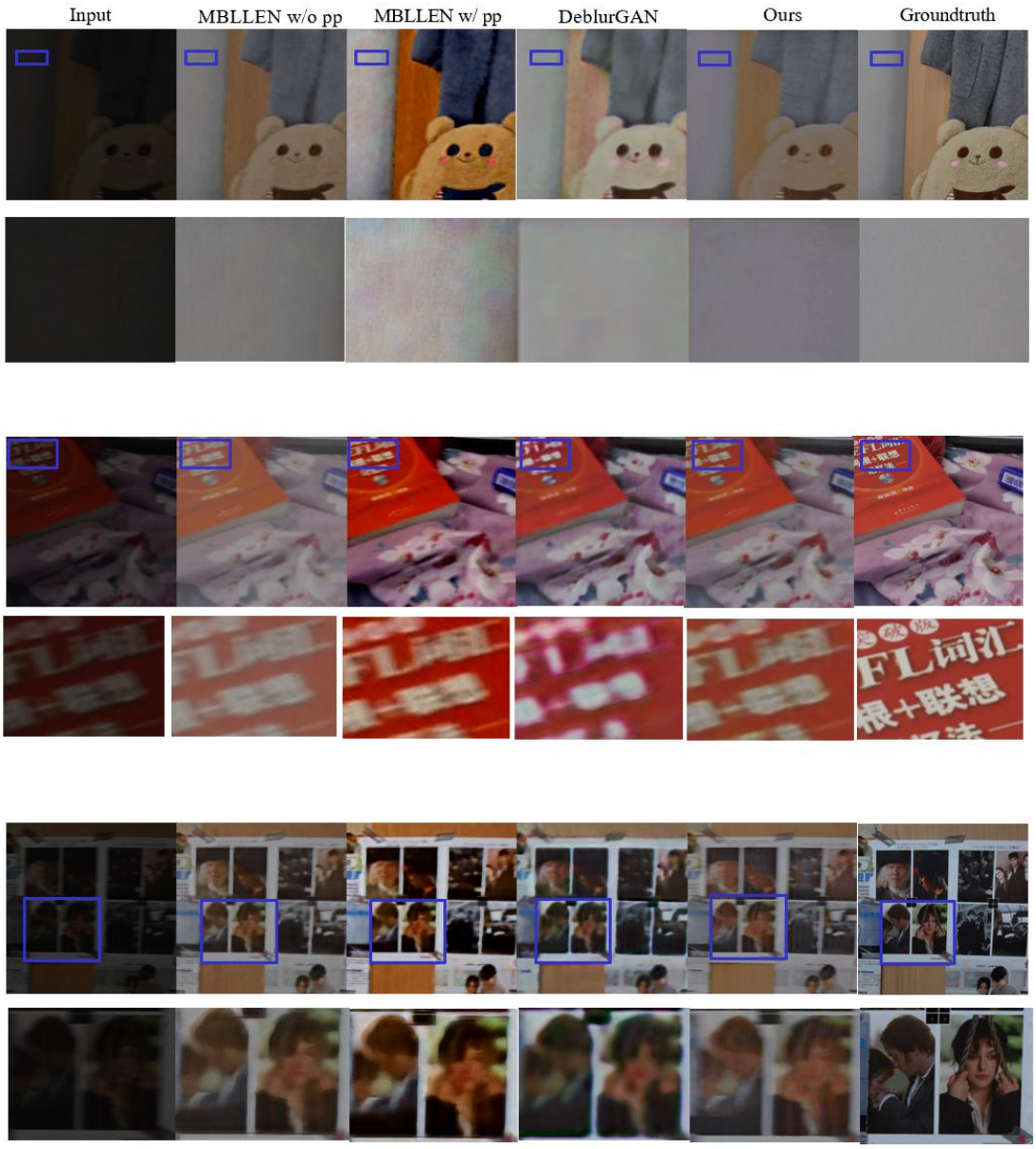

Fig. 2.12. Sample outputs with enlarged details for different network architecture. The first column are the input images, and the last column are ground truth images. The odd rows are images of size 512×512, and the even rows are images that zoomed into details.

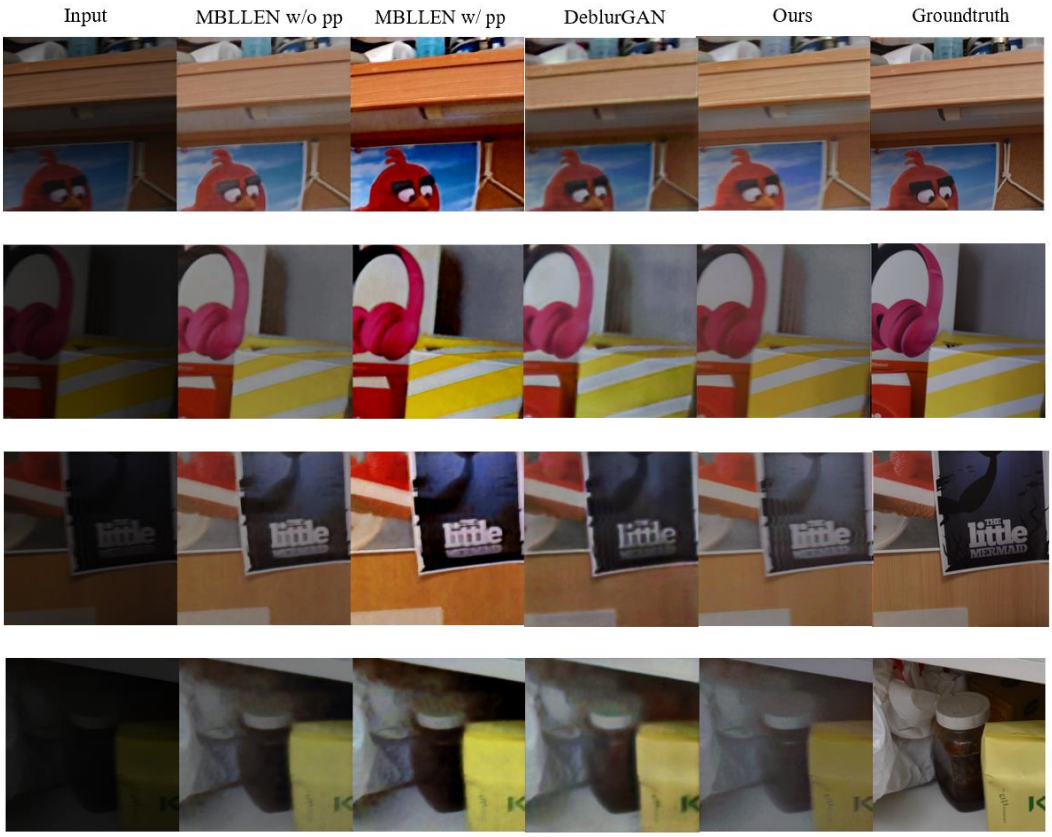

Fig. 2.13. More sample outputs details for different network architecture. The first column are the input images, and the last column are ground truth images.

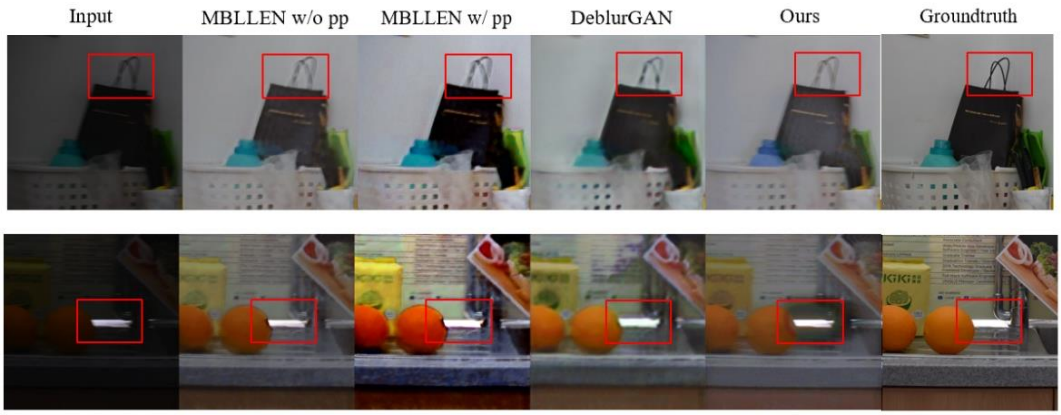

Fig. 2.14. Sample failed cases for different network architecture.

2.3.3 Ablation Studies

We conduct an ablation study to demonstrate which part of our approach contributes the most to the results. All different architectures are generally trained for 30 epochs with the same parameters, and results of the testing set of our dataset are reported in each group of ablation studies.

2.3.3.1 Generator Module

We first conduct a set of experiments to compare the performance of different network designs for the Generator module. We test 4 different design: U-Net, U-Net + ResNeXt, U-Net + Bilinear U-Net + ResNeXt + Bilinear. We show results on the testing set of our datasets in Table 2.2. Note that to better demonstrate the performance, the Discriminator module is not implemented.

	ResNeXt	**Bilinear**	**PSNR**	**SSIM**
U-Net	×	×	18.68	0.586
	√	×	19.23	0.596
	×	√	21.03	0.614
	√	√	**21.53**	**0.629**

Table 2.2. Network performance with different network designs for the generator.

From the table, we can see that ResNeXt slightly helps improve the numerical performance, and when bilinear is not employed for the up-sampling step in the U-Net architecture, the performance of our network drops significantly.

2.3.3.2 Critic (Discriminator) Module

We conduct a set of experiments to investigate the effectiveness of the Critic module. We test different settings on our datasets, including learning without Critic, with Critic and WGAN loss, and with Critic and WGAN-GP loss. The results on the testing sets are shown in Table 2.3.

	Critic	**Loss**	**PSNR**	**SSIM**
U-Net + ResNeXt + Bilinear	×	-	21.53	0.629
	√	WGAN	22.33	**0.651**
	√	WGAN-GP	**22.58**	0.648

Table 2.3. Network performance with different network designs for the critic.

We can see that the critic module significantly improves the numerical performance for both evaluation metrics. For the loss function of WGAN and WGAN-GP, the difference between their best final performance is minor. However, WGAN-GP does a good job of stabilizing the training process and, as a result, reduces the training time. Details of training procedure about the evaluation metrics are shown in Fig. 2.15 and Fig. 2.16.

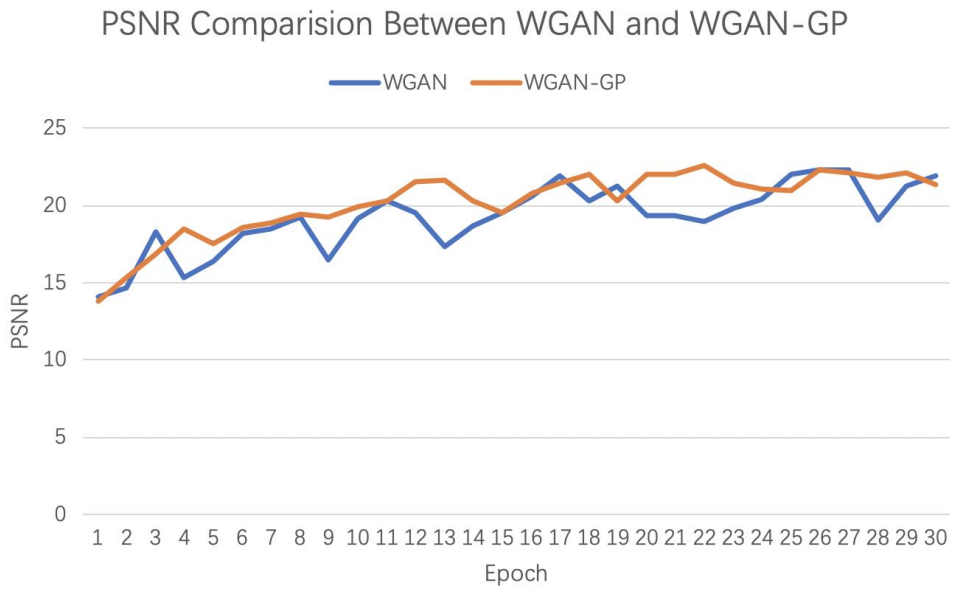

Fig. 2.15. Network training performance of PSNR for WGAN and WGAN-GP loss.

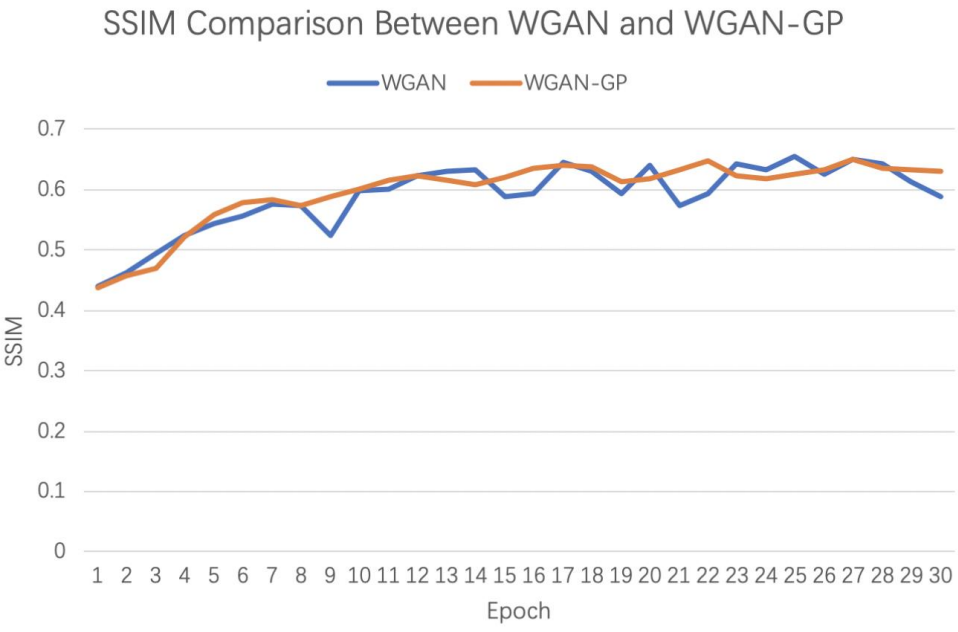

Fig. 2.16. Network training performance of SSIM for WGAN and WGAN-GP loss.

2.3.3.3 Fine-Tune State

Lastly, we conduct a compared experiment to investigate the effectiveness of the fine-tune state with *CoBi* loss. For the fine-tune state, apart from the basic 30 epochs, we fine-tune the model for another 20 epochs for convergence. Specifically, one experiment is to fine-tune the model with perceptual + L_1 + *CoBi* loss for the generator module, and another is to continue to train the model with the previous perceptual + L_1 loss. The fine-tune results on the testing sets are shown in Table 2.4.

	Fine-Tune	**PSNR**	**SSIM**
WGAN-GP	×	22.87	0.661
	√	**23.56**	**0.679**

Table 2.4. Network performance with or without fine-tuning with *CoBi* loss.

From the numerical result, we can see that the fine-tuning state with CoBi loss is essential to better restore a blurry image, because CoBi loss takes into account the spatial constraint to generate more structured images.

2.4 Evaluation

2.4.1 Data Collection

The dataset we collect meets our expectations. The average intensity of all the frames is around 20 to 30, which is good enough for us to do training using deep learning. Most importantly, our data collection method solves one very important issue and difficulty in this area that ground truth information for low-light video is hard to obtain.

2.4.2 Frame Enhancement & Video Enhancement

For video enhancement, we cut the original video into frames and do framewise enhancement; then we combine the images together to reconstruct the video. After turning the video into image frames and pre-processing the frame images, we can align our input images with our taken ground truth correspondingly. With these data, we are able to train a deep learning model. In total, our dataset contains 1,278 input and ground truth pairs of size 512×512×3.

Our testing results show that the frame enhancement is somehow successful, but still, there are gaps between frame enhancement and video enhancement.

One major concern is that the output image sequence may be inconsistent in brightness, contrast, etc., making the resulting video to be inconsistent temporally. However, in practice, we find that the output image sequence is consistent in the time domain, so we directly use the output image sequences to form the videos. This consistency could contribute to the consistency in the input sequence. Since our inputs are video frames, they are inherently consistent in the time domain. After passing through the same model, the consistency between adjacent frames still resists.

Another concern is that there may be some visual problems around the concatenating pixels between different frame fragments. In practice, such a visual problem does exist, so we may need some further image post-processing technique to resolve this concatenating problem.

3. Discussion

3.1 RGB Data vs. Raw Data

One of the main contributions of our project is to provide a model that can run directly on regular data format (i.e., ".mp4", ".mov"). Without the requirement of raw data from the camera, it undoubtedly widens the application field. Theoretically, the raw data is not compressed, with each pixel represented by 36 bits, while RGB data has only 24 bits for each pixel. Therefore, the information provided by raw data to the neural network is richer, and therefore using RGB data to train a model will suffer in this aspect. This may partially explain why our model is inferior to the state-of-the-art models trained on raw data in terms of evaluation metrics such as PSNR and SSIM.

3.2 Real Data vs. Synthetic Data

In our project, we use real data to train our model. Synthetics data are also widely used in image enhancement because the input and ground truth data pairs are difficult to get. For example, in [18], they use synthetic low-light images and videos to train the model. Synthetics data solves the problem that capturing real-world low-light images with ground truth is difficult. However, the synthetic data may be quite different from real data in nature. In a recent work on synthetic noise [19], researchers found that robust learning methods that work well on synthetic noise may not work as well on real-world noise. We are driven by this result to collect real data and find a way to capture the ground truth.

3.3 Data Generalization and Analysis

On the positive side, our proposed dataset contributes to this problem in two ways, as just mentioned. Firstly, our data better simulates the data distribution in the real world than synthetic data, given the model stronger generalization ability. Secondly, we use RGB data as model input so that there is a lower threshold considering the application scenario. Our model can be applied to the enhancement of surveillance video or medical imaging without requiring the raw data from the camera.

Admittedly, our dataset does have limitations in terms of diversity and scale. Regarding the scale, our final model is trained on a dataset of 1,278 images in total, each has a 512×512 resolution, which means many real-world scenarios are not included in our dataset. Furthermore, there is a lack of diversity with respect to dynamic motion because the way we create dynamic scenes is to let the camera move, instead of moving certain objects while keeping the other static. All the objects in the captured scenes are moving. Hence, the dataset may not represent the situations where there is a big difference between the trajectories of each moving object.

3.4 Limitation in Consistency

There are two levels of inconsistency: spatial and temporal. On the spatial level, since we cut the original frame into 512×512 fragments and each fragment is trained independently, inconsistency may exist at the junction point when we concatenate the 512×512 fragments as a large frame, as shown in Fig. 3.1. Using a large input size in the training process may solve this problem, but it requires much more memory and computing capacity, which is unrealistic in our case. On the temporal level, our result cannot guarantee a seamless connection between adjacent frames after we concatenate the enhanced frames. On the time dimension, fine-tuning using time series methods may be necessary. We expect future research will yield more consistent results on the basis of our experimental result.

Fig. 3.1. Restored video though frame fragment concatenation.

3.5 Limitation in Deblurring Functionality

We design the model with the initial objective of both brightening and deblurring. During the experiment, the model achieves satisfactory results in brightening, with good visual quality and color authenticity. However, the shortcoming in the output is a lack of sharp edges and details. We speculate the reasons are as follows.

Firstly, the size of the dataset is not large enough for a complete convergence. There are many random factors that produce the motion blur in a dynamic scene. Without enough data to support training, it is hard for the model to learn a set of parameters that is applicable to all kinds of blurs. We observe that the loss stops declining after 50 epochs in the training process, suggesting that the network cannot extract enough information for further convergence.

Secondly, the model has some deficiencies in its own network architecture. The U-Net structure does not have a strong capability to reconstruct fine details. Many networks specially designed for deblurring in the literature either contain sub-nets for blur kernel estimation or uses multi-scale architecture. Multi-scale architecture means loss calculation is carried out at each scale so that the network can deal with small amounts of blur restoration in each scale. Since our project also takes the low-light scene enhancement into account, we avoid over-complex structures targeted for deblurring.

4. Conclusions

In this project, we introduce a novel method to collect data for low-light video enhancement and make the collected dataset publicly available. We also propose a deep learning approach to solve this task end-to-end. Experimental results show that our proposed network achieves the most state-of-the-art performance on our dataset both quantitatively and qualitatively.

4.1 Summarization of Our Work

At the very early stage, we reviewed many state-of-the-art methods in the area of denoising, deblurring, super-resolution, and low-light enhancement for both images and videos. We summarized their techniques and limitation of their works, then came up with our project of low-light video enhancement using deep learning, together with our two initial objectives:

1) Solve the current difficulty in capturing ground truth videos in low-light conditions.

2) Propose a deep learning model that can enhance low-light videos.

With our novel data collection mechanism, we collected our own dataset and trained our model on the dataset. This process was not very smooth, because most of the data we collected for the first time was found to be useless for training and testing. This is because the input RGB images were too, so the pixel contained very limited information to be. We then recollected our data and tested them again. During model training, we had found that the frames could be lightened up, but it is still blurred. Through testing numerically, though our model reaches state of the art among previous work, when testing perceptually, we found that the result frames are brighter but still had some minor noise and motion blur.

4.2 Future Direction

As mentioned in the last section, denoising and deblurring functionality are limited in our model. This could be caused by several reasons:

- Improper architecture design
- Improper loss function design
- Limited dataset (only over 1,000 samples)
- Insufficient information in RGB images

These could be future directions for further research. For example, as our dataset is very limited, further work could be done to collect more datasets, so the network may be getting more robust to deblurring function.

Limited by the memory size, we divide the original inputs into patches during training. The final restored videos are generated through concatenating the frame fragments. By doing so, inconsistency around the concatenating edges exists. So, another possible research direction is to come up with an end-to-end network that can restore videos without such concatenating inconsistency (e.g., through operation on high-resolution inputs).

5. References

[1] S. Su, M. Delbracio, J. Wang, G. Sapiro, W. Heidrich, and O. Wang, “Deep Video Deblurring for Hand-Held Cameras,” in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, Jul. 2017, pp. 237–246.

[2] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas, “DeblurGAN: Blind Motion Deblurring Using Conditional Adversarial Networks,” in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, Jun. 2018, pp. 8183–8192.

[3] Q. Fu, X. Di, Y. Zhang, “Learning an Adaptive Model for Extreme Low-light Raw Image Processing,” *arXiv:2004.10447v1 [eess.IV]*, Apr. 2020.

[4] C. Chen, Q. Chen, J. Xu, and V. Koltun, “Learning to See in the Dark,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, Jun. 2018, pp. 3291–3300.

[5] S. Ai and J. Kwon, “Extreme Low-Light Image Enhancement for Surveillance Cameras Using Attention U-Net,” *Sensors*, vol. 20, no. 2, p. 495, Jan. 2020.

[6] F. Lv, Y. Li, F. Lu, “Attention Guided Low-light Image Enhancement with a Large Scale Low-light Simulation Dataset,” *arXiv:1908.00682v3 [eess.IV]*, Mar. 2020.

[7] Z. Fu, Y. Zheng, H. Ye, Y. Kong, J. Yang, and L. He, “Edge-Aware Deep Image Deblurring,” *arXiv:1907.02282 [cs.CV]*, Jul. 2019.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville. and Y. Bengio, “Generative Adversarial Nets,” *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[9] O. Kupyn, T. Martyniuk, J. Wu, and Z. Wang, “DeblurGAN-v2: Deblurring (Orders-of-Magnitude) Faster and Better,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South), Oct. 2019, pp. 8877–8886.

[10] M. S. Rad, B. Bozorgtabar, U.-V. Marti, M. Basler, H. K. Ekenel, and J.-P. Thiran, "SROBB: Targeted Perceptual Loss for Single Image Super-Resolution," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South), Oct. 2019, pp. 2710–2719.

[11] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *arXiv:1701.07875 [cs, stat]*, Dec. 2017, Accessed: Apr. 28, 2020. [Online]. Available: http://arxiv.org/abs/1701.07875.

[12] P. Li, L. Zhao, D. Xu, and D. Lu, "Incorporating Multiscale Contextual Loss for Image Style Transfer," in *2018 IEEE 3rd International Conference on Image, Vision and Computing (ICIVC)*, Chongqing, Jun. 2018, pp. 241–245.

[13] X. Zhang, Q. Chen, R. Ng and V. Koltun, "Zoom to Learn, Learn to Zoom," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 3757-3765.

[14] M. Noroozi, P. Chandramouli, and P. Favaro, "Motion Deblurring in the Wild," in *German Conference on Pattern Recognition*, pp. 65–77, 2017.

[15] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, A. Courville, "Improved Training of Wasserstein GANs", in *arXiv:1704.00028 [cs.LG]*, Dec. 2017

[16] S. Xie, R. Girshick, P. Dollár, Z. Tu and K. He, "Aggregated Residual Transformations for Deep Neural Networks," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 2017, pp. 5987-5995.

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, Jun. 2009, pp. 248–255.

[18] F. Lv, F. Lu, J. Wu, C. Lim, "MBLLEN: Low-light Image/Video Enhancement Using CNNs" in *2018 The British Machine Vision Conference (BMVC)*, Newcastle, UK, Sep. 2018.

[19] L. Jiang, D. Huang, and W. Yang, "Synthetic vs Real: Deep Learning on Controlled Noise," arXiv:1911.09781 [cs, stat], Nov. 2019, Accessed: Apr. 29, 2020. [Online]. Available: http://arxiv.org/abs/1911.09781.

6. Appendix A: Meeting Minutes

6.1 Minutes of the 1st Project Meeting

Date: Sep 6, 2019
Time: 3:00-4:00 pm
Place: Room 3508
Present: All
Absent: None
Recorder: Yingshan

1. Approval of minutes
   This was the first formal group meeting, so there were no minutes to approve.
2. Report on progress
   2.1. All team members have read the instructions of the Final Year Project online and have done research for the topic.
   2.2. We discussed with our instructor about our topic selection.
3. Discussion items
   3.1. The goal of our project is basically to try to do something like converting a video captured in a dark environment, possibly with blurry scenes to a brighter and clear one.
   3.2. The scope of the project includes collecting datasets by manually capturing input-output pairs, designing neural networks to do image deblurring as well as perception evaluation.
   3.3. In the literature, there are already many images or video deblurring algorithms, whether or not we can apply their methods on our new dataset calls for further exploration.
4. Goals for the coming week
   4.1. All group members will study related papers provided by Prof. Chen.
   4.2. A specific focus of our project, as well as the data collection method, should be determined within a week.
5. Meeting adjournment and next meeting
   The meeting was adjourned at 4:00 pm.
   The next meeting will be at 4:00 pm on Sep 9 at the Library 1-352.

6.2 Minutes of the 2nd Project Meeting

Date: Sep 9, 2019
Time: 4:00-6:00 pm
Place: Library 1-352
Present: All
Absent: None
Recorder: Yingshan

1. Approval of minutes
   The minutes of last meeting were approved without amendment.
2. Report on progress
   2.1. All group members have studied papers provided Prof. Chen.
   2.2. Xuanyi found another paper talking about collecting input-output image pairs containing identical scene.
   2.3. Ka Leong came up with a method for capturing video with identical scenes using automatic turntable.
3. Discussion items
   3.1. Collecting input-output video pairs by shooting one video with the light off and another with light on might be a good idea for data collection. We will ask Prof. Chen's opinion before making further decisions.
   3.2. Using raw images acquired from Stereo camera may be too challenging, since the alignment between left and right images should be addressed at the same time with deblurring.
   3.3. Doing image processing in the dark by taking both RGB image and infrared image as input may be unrealistic, since RGB image and infrared image are so different that their information cannot complement each other.
   3.4. The group decided that our project should particularly focus on image deblurring rather than generally improving image quality.
4. Goals for the coming week
   4.1. All group members will drop by Prof. Chen's office and discuss our ideas with him.
   4.2. All group members will need to study and compare network architectures and data collection methods for image deblurring in the literature.
   4.3. All group members will think about the outline of the proposal.
5. Meeting adjournment and next meeting
   The meeting was adjourned at 6:00 pm.

The date and time of the next meeting will be on Sep 16, and the place will be set later.

6.3 Minutes of the 3rd Project Meeting

Date: Sep 16, 2019
Time: 8:30-10:30 pm
Place: Library LG3-04
Present: Xuanyi, Yanming, Yingshan
Absent: Ka Leong
Recorder: Yingshan

1. Approval of minutes
   The minutes of last meeting were approved without amendment.
2. Report on progress
   2.1. All group members have agreed that our data collection method is to capture input videos in low-light conditions and capture long-exposure images as ground truth. The number of ground truth image will be less than the number of video frames, but we will make sure that sufficient samples are collected before training.
   2.2. Xuanyi prepared her Nikon camera.
   2.3. Ka Leong bought an automatic turntable.
   2.4. Yanming studied how to extract every frame from a video through OpenCV.
   2.5. All group members decided to shoot videos by putting the camera on the turntable to simulate movement.
3. Discussion items
   3.1. We discussed whether to use auto mode or manual mode when capturing the ground truth.
      3.1.1. Auto mode is more convenient, but parameters like aperture size, shutter speed and ISO will change according to different scenes.
      3.1.2. In manual mode, we can adjust the parameters as we like, but it might be time-consuming.
   3.2. Each video will last for 3 second, which is corresponding to the turntable turning around 45 degrees.
   3.3. A ground truth image will be captured every 2 degrees.
   3.4. The outline of the proposal is finished during the meeting, each part is distributed to one group member.
4. Goals for the coming week

4.1. All group members will finish his/her part of the proposal
4.2. We will let Prof. Chen examine the proposal before submission.
4.3. All group members will study the neural network architectures for image deblurring and find some useful Github repositories.

5. Meeting adjournment and next meeting
   The meeting was adjourned at 10:30 pm.
   The date and time of the next meeting will be set later. Group members will be notified by WeChat.

6.4 Minutes of the 4th Project Meeting

Date: Sept 24, 2019
Time: 6:00-8:00 pm
Place: HKUST
Present: All
Absent: None
Recorder: Ka Leong

1. Approval of minutes
   The minutes of last meeting were approved without amendment.
2. Report on progress
   2.1. Xuanyi prepared her Nikon camera.
   2.2. Ka Leong brought his automatic turntable with the remote control.
   2.3. Ka Leong and Yingshan prepared the scenes to collect data.
   2.4. Xuanyi controlled the camera and was responsible for the camera setting.
   2.5. Yanming was in charge of the remote control
3. Discussion items
   3.1. We discussed the methods for data preprocessing and distribute the work
   3.2. We decided to divide the group into 2 small groups for further data collection, as two people should be enough for data collection, to improve the efficiency.
4. Goals for the coming week
   4.1. The two small groups will collect data when they are free.
   4.2. All group members will start doing the data preprocessing
   4.3. We will let Prof. Chen know our progress of data preprocessing.
5. Meeting adjournment and next meeting
   The meeting was adjourned at 8:00 pm.
   The date and time of the next meeting will be set later. Group members will be notified by WeChat.

6.5 Minutes of the 5th Project Meeting

Date: Oct 9, 2019
Time: 4:00-5:00 pm
Place: Library LG3-Garden
Present: All
Absent: None
Recorder: Ka Leong

1. Approval of minutes
   The minutes of last meeting were approved without amendment.
2. Report on progress
   2.1. All group members agreed the criteria of dataset, i.e., the average intensity of the video frames needs to be larger than 20.
   2.2. All group members sit together to select the useable data.
3. Discussion items
   3.1. We discussed the details of preprocessing about the further steps:
      3.1.1. Find input & ground truth pairs
   3.2. We discussed the method for designing a native model in Pytorch.
   3.3. We distribute the work to all the group members.
4. Goals for the coming week
   4.1. All group members will start their own work (preprocessing and model design).
   4.2. All group members will start working on their individual essay.
5. Meeting adjournment and next meeting
   The meeting was adjourned at 5:00 pm.
   The date and time of the next meeting will be set later. Group members will be notified by WeChat.

6.6 Minutes of the 6th Project Meeting

Date: Oct 28, 2019
Time: 7:00-9:00 pm
Place: Library LG4-08
Present: All
Absent: None
Recorder: Ka Leong

1. Approval of minutes

The minutes of last meeting were approved without amendment.

2. Report on progress
   2.1. All group members sit together to finish the monthly report
   2.2. Yingshan contact Prof. Chen for signature for the monthly report.
   2.3. Yingshan, Yanming, and Ka Leong showed the research on some advanced papers.
   2.4. Xuanyi showed her preliminary results on the training model.
3. Discussion items
   3.1. We discussed the possible reasons of the bad result of the preliminary model.
   3.2. We discussed about some possible alternatives for other advanced models to improve the performance of our model.
4. Goals for the coming week
   4.1. All group members will continue to find out the possible reasons for the bad result.
   4.2. We will try to schedule a timeslot to have a meeting with Prof. Chen
5. Meeting adjournment and next meeting
   The meeting was adjourned at 9:00 pm.
   The date and time of the next meeting will be set later. Group members will be notified by WeChat.

6.7 Minutes of the 7th Project Meeting

Date: Nov 19, 2019
Time: 2:30-4:30 pm
Place: WeChat Voice Call
Present: All & Prof. Chen
Absent: None
Recorder: Ka Leong

1. Approval of minutes
   The minutes of last meeting were approved without amendment.
2. Report on progress
   2.1. Prof. Chen pointed out that there should be some bugs inside our training models, we need to spend some time for debugging
   2.2. Prof. Chen gave us some suggestions and provided some papers for reference.
3. Discussion items
   3.1. We discussed the distribution of work, like the debugging and researching.

3.2. We discussed how to work virtually at home for the remaining Fall semester and the coming Winter semester.

4. Goals for the coming week
   4.1. All group members will work on the debugging of the network.
   4.2. All group members will continue their research on possible suitable papers
5. Meeting adjournment and next meeting
   The meeting with Prof. Chen was adjourned at 3:30 pm.
   The continued meeting was adjourned at 4:30 pm.
   The date and time of the next meeting will be set later. Group members will be notified by WeChat.

6.8 Minutes of the 8th Project Meeting

Date: Dec 11, 2019
Time: 11:00 am-12:00 pm
Place: Prof. Chen's office room
Present: All & Prof. Chen
Absent: None
Recorder: Ka Leong

1. Approval of minutes
   The minutes of last meeting were approved without amendment.
2. Report on progress
   2.1. Xuanyi made a PowerPoint to show our work to Prof. Chen.
   2.2. Prof. Chen pointed out some possible improvement for image matching: image registration and image mask for loss, as the current frame match is not perfect.
3. Discussion items
   3.1. We discussed the method of realization of the image registration and image mask for the loss.
   3.2. We discussed how to distribute the work to all the group members.
4. Goals for the coming week
   4.1. Ka Leong and Xuanyi will finish the image registration coding for preprocessing.
   4.2. Yingshan and Yanming will finish the research on image mask for loss computation and realize it into real code.
5. Meeting adjournment and next meeting
   The meeting with Prof. Chen was adjourned at 12:00 pm.

The date and time of the next meeting will be set later. Group members will be notified by WeChat.

6.9 Minutes of the 9th Project Meeting

Date: Jan 13, 2020
Time: 3:00-6:00 pm
Place: Room 3464
Present: All & Communication Tutor
Absent: None
Recorder: Ka Leong

1. Approval of minutes
   The minutes of last meeting were approved without amendment.
2. Report on progress
   2.1. The communication tutor helped us to go over the progress report and draw the logic diagram for us.
   2.2. The communication tutor gave us some suggestion for the progress report.
3. Discussion items
   3.1. We discussed some existing problems in the proposal report.
   3.2. We come up with some alternatives to improve for the writing in the progress report.
4. Goals for the coming week
   4.1. All group members will continue to work on the remaining work assigned during the last meeting
   4.2. All group members will start to write the progress report.
5. Meeting adjournment and next meeting
   The meeting with the communication tutor was adjourned at 6:00 pm.
   The date and time of the next meeting will be set later. Group members will be notified by WeChat.

6.10 Minutes of the 10th Project Meeting

Date: Feb 24, 2020
Time: 8:00-9:00 pm
Place: WeChat Voice Call
Present: All
Absent: None
Recorder: Ka Leong

1. Approval of minutes
   The minutes of last meeting were approved without amendment.
2. Report on progress
   2.1. Yingshan pointed out that we need to do more research on different loss.
   2.2. Xuanyi pointed out that we need to add another pre-trained deblur model on top of our model output
   2.3. Yanming determined to use PSNR and SSIM as our evaluation metrics for comparison.
   2.4. Ka Leong finished the integration of the codes.
3. Discussion items
   3.1. We discussed about the structure of the progress report.
   3.2. We discussed about the work distribution within the group.
4. Goals for the coming week
   4.1. All group members will work on the writing of progress report and finish it before the deadline
   4.2. All group members need to continue to finetune the parameters for different models
5. Meeting adjournment and next meeting
   The meeting with the communication tutor was adjourned at 9:00 pm.
   The date and time of the next meeting will be set later. Group members will be notified by WeChat.

6.11 Minutes of the 11th Project Meeting

Date: Feb 28, 2020
Time: 6:00-6:30 pm
Place: WeChat Voice Call
Present: All
Absent: None
Recorder: Ka Leong

1. Approval of minutes
   The minutes of last meeting were approved without amendment.
2. Report on progress
   2.1. All the group members almost finished about the progress report writing
   2.2. All the group members finished 80% of the parameter finetuning.
3. Discussion items

3.1. We discussed about future work in the coming two months in details.
   3.1.1. Reduce the cut size to 256
   3.1.2. Add the dataloader to WGAN model
   3.1.3. Add some constant value first to the input frames
   3.1.4. Try the 3d convolution
   3.1.5. Concatenate our current model with another deblur network
   3.1.6. Need to collect more data after getting back to school
3.2. We discussed about to re-write the GRANT Chart in the progress report.

4. Goals for the coming week
   4.1. All group members will work on the future work accordingly.
   4.2. Contact Prof. Chen for some further advise.
5. Meeting adjournment and next meeting
   The meeting with the communication tutor was adjourned at 9:00 pm.
   The date and time of the next meeting will be next Friday.

6.12 Minutes of the 12th Project Meeting

Date: Apr 21, 2020
Time: 5:30-6:30 pm
Place: Zoom meeting
Present: All & Communication Tutor
Absent: None
Recorder: Xuanyi

1. Approval of minutes
   The minutes of last meeting were approved without amendment.
2. Report on progress
   2.1. Communication tutor gave comments on the progress report in Feb.
   2.2. Communication tutor helped us on the structure of our project and gave suggestions for the final report.
3. Discussion items
   3.1. We discussed some existing problems in the progress report. CT pointed out some parts that we should write more clearly and parts that are redundant.
   3.2. We come up with some alternatives to improve for the writing in the final report.
4. Goals for the coming week
   4.1. All group members will continue to work on the remaining work assigned during the last meeting.
   4.2. All group members will start to write the final report.

7. Appendix B: Progress

7.1 Distribution of Work

Task	Ka Leong Cheng	Xuanyi Li	Yanming Kang	Yingshan Chang
Do the Literature Survey	○	○	○	●
Proposal Writing	●	○	○	○
Video Data Collection	○	●	○	○
Long Exposure Ground Truth Collection	○	●	○	○
Data Pre-processing	○	○	●	○
CNN Architecture Design	●	○	○	○
Loss Function Design	●	○	○	○
CNN Implementation	○	○	○	●
Monthly Report Writing	○	●	○	○
Network Training	○	○	●	○
Architecture Adjustment (According to problem discovered later)	○	○	○	●
Second-stage data collection	○	●	○	○
Further image registration between input and ground truth	●	○	○	●
Loss Function design	○	●	○	○
Network Tuning	○	○	●	○
Experiments with different model architecture	○	○	○	●
Experiments with different loss	○	●	○	○
Testing with numerical metrics (PSNR, SSIM)	●	○	○	○
Progress Report Writing	○	○	●	○
Testing	●	○	○	○
Perception Evaluation	○	○	●	○
Ablation Study	○	●	○	○
Final Report Writing	●	○	○	○
Presentation Preparation	○	○	○	●
Project Poster Design	○	●	○	○

● Leader ○ Assistant

7.2 GANTT Chart

Task/Month	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May
Literature Survey	■								
Proposal Writing	■								
Data Collection	■								
Data Preprocessing		■	■						
Architecture Design		■	■	■					
Loss Function Design		■	■	■	■	■	■		
Model Implementation		■	■	■	■	■	■		
Monthly Report		■	■	■					
Network Training					■	■	■	■	
Network Adjustment						■	■	■	
Network Tuning						■	■	■	
Progress Report					■	■			
Testing							■	■	
Human Perception								■	
Ablation Study								■	
Evaluation								■	
Final Report								■	
Presentation									■
FYP Video									■

7.3 Hardware

PCs

GPU (from Google Colaboratory)

Nikon Camera (with remote control function)

Automatic Rotating Turntable (with remote control function)

SIDO All In 1 Card Reader

Hard Disk (for storage)

7.4 Software

GitHub

Python

MATLAB

Jupyter Notebook

Anaconda

8. Appendix C: Sample data

The full dataset includes the raw video, ground truth images, which can be accessed: https://drive.google.com/drive/folders/10eAamNZV0VQkg_hRYi4ulxbglFKjxPWD?usp=sharing.