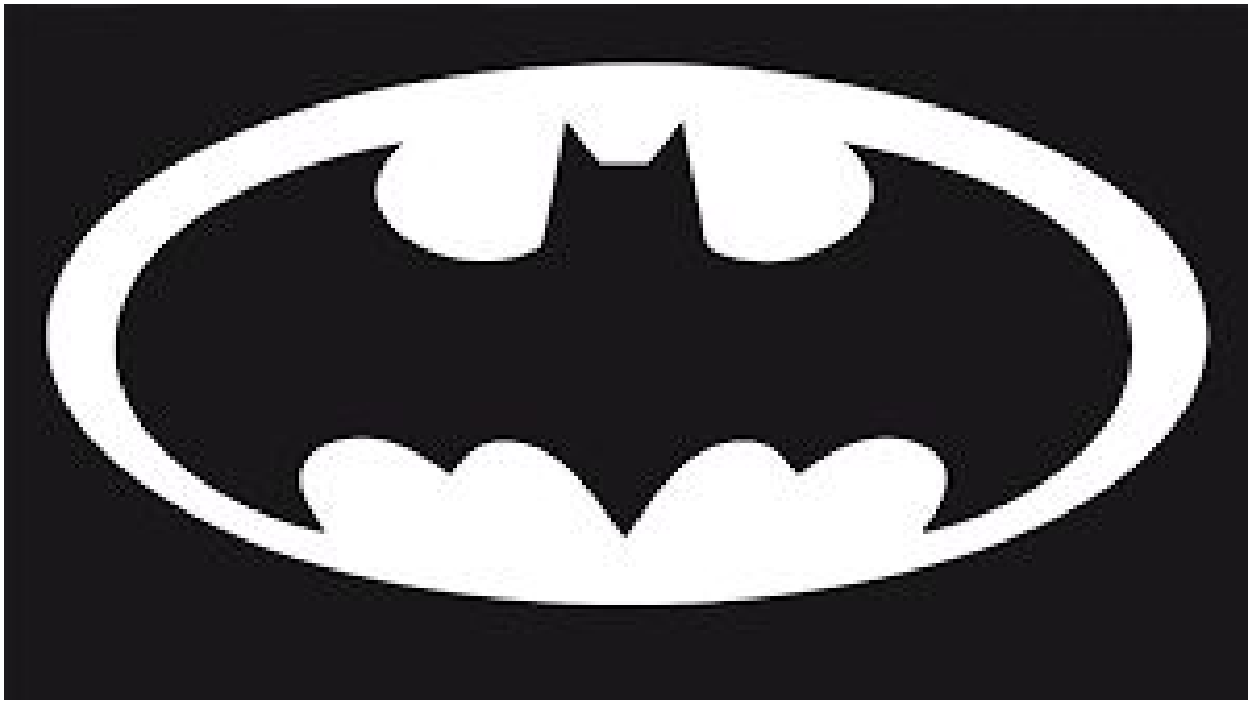
Computer Vision

Be a Batman - See Motion in the Dark

R08921040	R08942087	R08921098	R08921115
徐均筑	吳彬睿	蔣思揚	林修同
NTU EE	NTU GICE	NTU EE	NTU EE



1. Introduction

There is some extremely low-light circumstance in our daily life, like candle dinner, outdoors under the moonlight and so on. In this regime, the traditional camera processing pipeline breaks down. Although researchers have proposed techniques for denoising, deblurring, and enhancement of low-light images. These techniques generally assume that images are captured in somewhat dim environments with moderate levels of noise. In recent work[2], Chen et al. addressed this problem and proposed a siamese network, which gives to impressive results. However, they only consider one frame at a time during

inference. Intuitively, taking the temporal correlations of consecutive frames into consideration is helpful. Therefore, we propose two methods, using CLSTM and 3D CNN, to take advantage of this useful information and obtain promising result compared with traditional pipelines.

2. Related work

Chen et al.[1] first proposed a new image processing pipeline that addresses the challenges of extreme low-light photography via a data-driven approach and train deep neural networks to learn the image processing pipeline for low-light raw data, including color transformations, demosaicing, noise reduction, and image enhancement. The pipeline is trained end-to-end to avoid the noise amplification and error accumulation that characterize traditional camera processing pipelines in this regime. In our work we want to get clear videos from processing the extremely low-light videos, but in [1], it only consider spatial artifacts but not temporal artifacts.

And in [2], Chen et al. proposed a siamese network that preserves color while significantly suppressing spatial and temporal artifacts. The model was trained on static videos only but was shown to generalize to dynamic video.

3. Our Methods

3.1 Model

As the input frames are continuous in the temporal dimension, taking the temporal correlations of these frames into consideration is intuitive and presumably helpful. In terms of achieving this goal, both the 3D CNN and the CLSTM are competent. Therefore, our model has two versions, which employ the 3D CNN and the CLSTM respectively.

3.1.1 Spatial feature extraction network

Spatial feature extraction is the key to the performance and processing speed. In our work, we use a structure akin to [3], where the network contains an encoder, a decoderand a multi-scale feature fusion module (MFF). We show the details of our spatial feature extraction network in Fig. 1(left). The encoder can be any 2D CNN model. Due to resource limitation, we apply a shallow ResNet-18 model as the encoder. The decoder

employs four up-projection modules to improve the spatial resolution and decreases the number of channels of the feature maps. The MFF module is designed to integrate features of different scales.

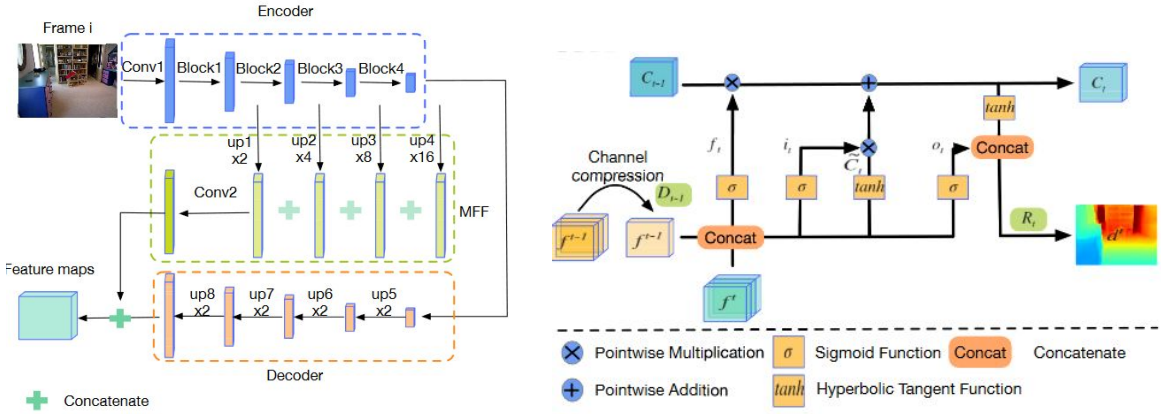


Figure 1. The structure of spatial feature extraction network (left) and CLSTM[3] (right).

3.1.2 CLSTM[3]

The structure of CLSTM is shown in Fig. 1(right). Specifically, the proposed CLSTM cell can be expressed as:

$$
\begin{aligned}
f_t &= \sigma\left(\left[f^t, D_{t-1}(f^{t-1})\right] * W_f + b_f\right), \\
i_t &= \sigma([f^t, D_{t-1}(f^{t-1})] * W_i + b_i), \\
\tilde{C}_t &= tanh([f^t, D_{t-1}(f^{t-1})] * W_C + b_C), \\
C_t &= f_t \times C_{t-1} + i_t \times \tilde{C}_t, \\
o_t &= \sigma([f^t, D_{t-1}(f^{t-1})] * W_o + b_o),
\end{aligned}
$$

where $*$ is the convolutional operator. W_f, W_i, W_C, W_o and b_f, b_i, b_C, b_o denote the kernels and bias terms at the corresponding convolution layers. After extracting the spatial features of video frames, we concatenate f_{t-1} with the feature map of current frame f_t to formulate a feature map with $2c$ channels. Next, we feed the concatenated feature map to CLSTM to update the information stored in memory cell. Finally, we concatenate the information in the updated memory cell C_t and the feature map of output gate, then feed them to next layer of CLSTM , continued until last output layer than obtain our final results.

3.1.3 Our Model 1 : Encoder + MFF + CLSTM

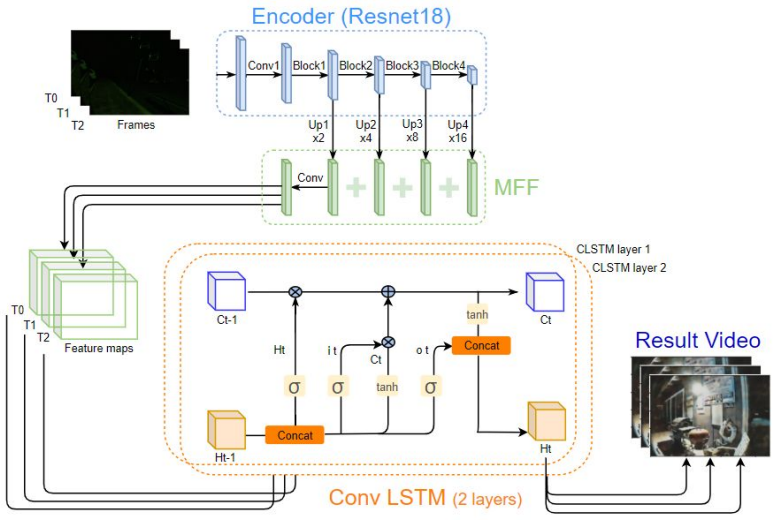


Figure 2. The structure of Model 1.

The first version of our model is shown in Fig. 2. We use a spatial feature extraction network to obtain feature maps for each frame and then feed them into CLSTM to capture long and short term temporal dependencies. Note that the decoder is discarded in this version because of hardware resource limitation.

3.2 Our Model 2 : Autoencoder + MFF + 3D CNN

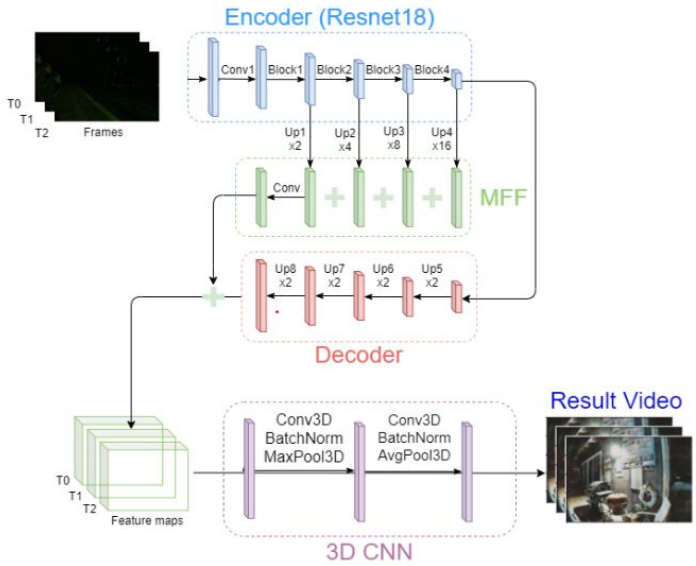


Figure 3. The structure of Model 2.

4. Experiments

4.1 Raw dark video dataset

We use the dataset of [2] which is collected by using a Sony RX100 VI camera, that can capture raw image sequences at approximately 16~18 frames per second in continuous shooting mode, and the buffer can keep around 110 frames in total. This is equivalent to 5.5 seconds video with 20 fps. The resolution of the image is 3672 X 5496. The dataset include indoor and outdoor scenes.

Because it is difficult to get the ground truth of extremely low-light dynamic videos, Chen et al. [2] collected both static videos with corresponding long-exposure images as their ground truth and dynamic videos without ground truth which is used only for perceptual experiments. Most scenes in the dataset are in the 0.5 to 5 lux range . And the dataset is proved having bias noise compared with the prediction by synthetic model which is applied to the ground truth by Chen et al.[2].

This dataset has 202 static videos for training and quantitative evaluation. Randomly divide them into approximately 64% for training, 12% for validation, and 24% for testing. Videos for the same scene are distributed within one of the sets but not across these sets. And Some scenes are in different lighting conditions. Examples are shown in Figure 4.

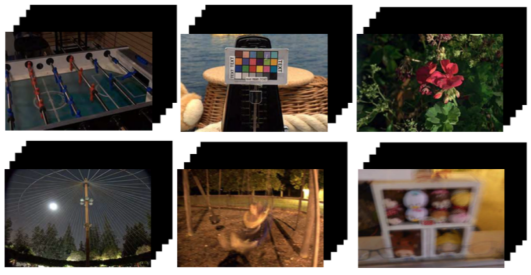


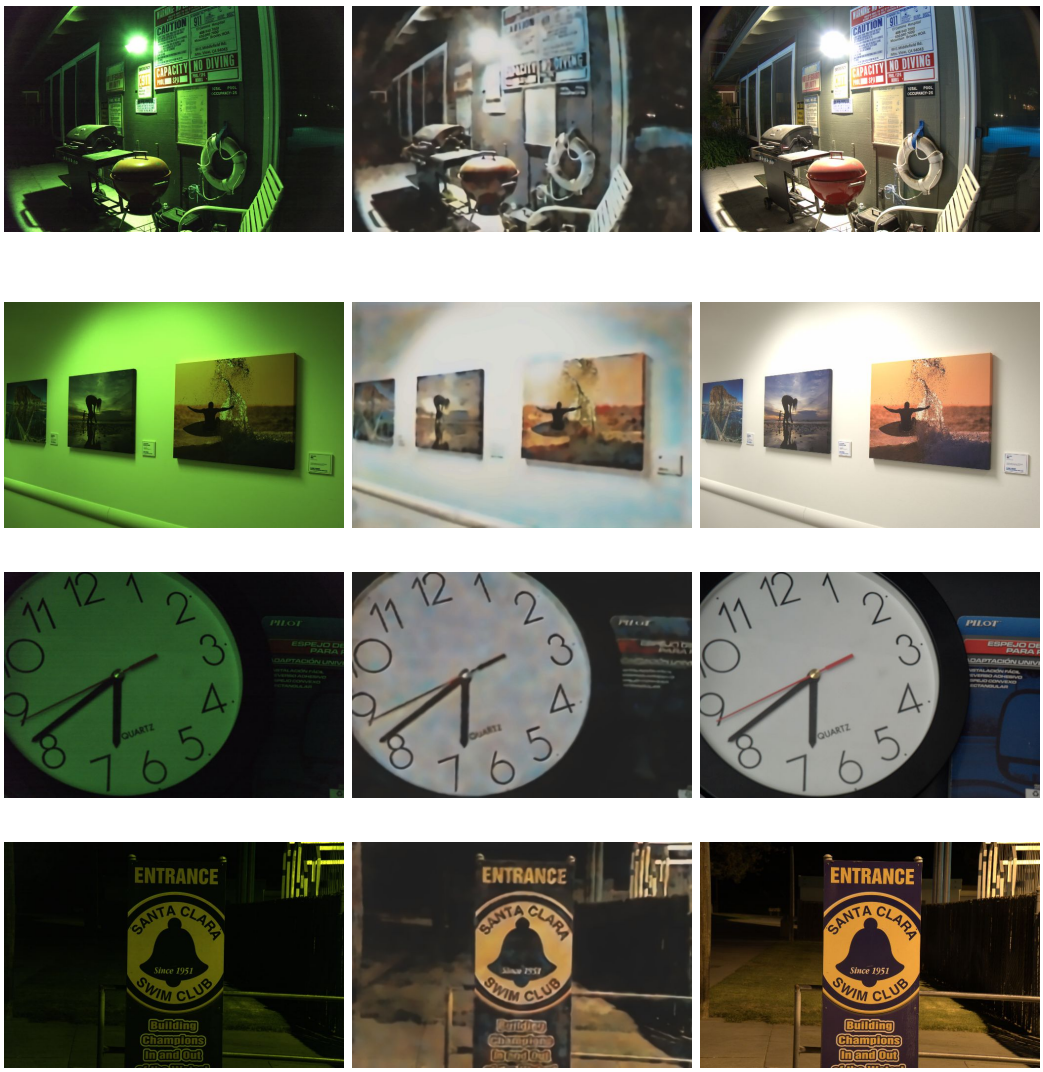
Figure 4. The first image is the long-exposure reference image (ground truth), the latter frames is the short-exposure images, which are dark in extremely low-light condition.

4.2 Training

Our method is implemented using Pytorch. We train our model on an Nvidia GTX 1080Ti GPU with 12 GB of memory. We use the L1/L2 loss and the AdamW optimizer, setting the batch size to 2. The initial learning rate is 10^{-4}. We keep training the network until validation loss doesn't improve for 3 epochs.

4.3 Result

Raw+VBM4D Our Result (Model 1) GroundTruth



Figures 5.

Figures 5. shows that the lighter circumstances are the more precise result we can get by our model. Otherwise, the results will contain more artifacts.

4.3.1 Youtube video demo (https://www.youtube.com/watch?v=jp6JZnTpg9k)

4.4 Image Quantitative Analysis

We follow the evaluation of different methods in [2] on the static test videos. The 5th frame of the output video is compared with ground truth using Peak Signal to Noise Ratio (PSNR) and the Structure Similarity Index (SSIM). And the long-exposure raw images are processed by Rawpy to form the sRGB ground truth. The average results over the entire test set are listed in Table 1.

Frame Quality evaluation

	PSNR (dB)	SSIM
Input+Rawpy	12.94	0.165
VBM4D+Rawpy	14.77	0.315
KPN+Rawpy	18.81	0.540
SMID [2]	**28.26**	**0.815**
CLSTM	**18.42**	0.593
3D-CNN	17.18	**0.615**

Table 1. Quantity evaluation of image quality on the static dataset

From Table 1, we can discover that although our experimental results are not comparable with SMID[2], our models still beat the traditional pipeline which takes preprocessed (with spatial and temporal denoising) raw RGB images as input.

From Figure 6., we can find that CLSTM and 3DCNN perform better than traditional pipeline. On the other hand, due to the GPU memory limitation, our model use L1 loss/ MSE (CLSTM) rather than perceptual loss (using VGG) and our original raw RGB images are resized. Hence, our results have more artifacts than SID [1] and SIMD [2].

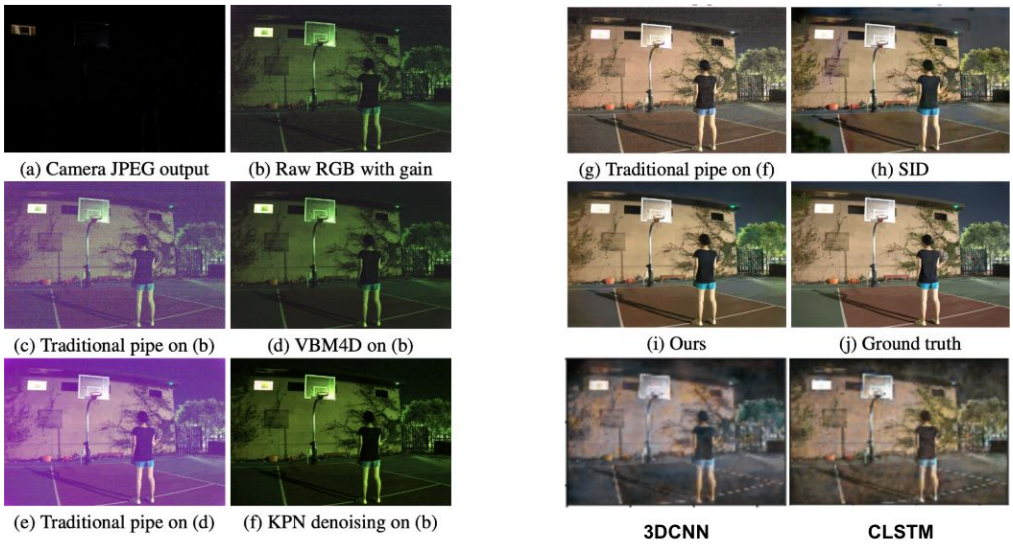


Figure 6.

4.5 Video Quality Analysis

We further evaluate the video quality of our model and also SID [1] and SMID [2]. We adopt the methodology of [2] that measures temporal error on every pair of consecutive frames using PSNR, TSSIM, and mean absolute error (MAE) on the static test videos. And we use terms in [2], such as temporal PSNR (TPSNR), temporal SSIM (TSSIM), and temporal MAE (TMAE) to distinguish the temporal variants from single-image metrics.

From Table 2., we can discover that our 3D-CNN model has better performance than SID [1] on temporal consistency, but still not surpass SIMD [2] yet.

Temporal consistency evaluation

	TPSNR (dB)	TSSIM	TMAE (x100)
SID [1] w/o VBM4D	33.72	0.939	1.56
SMID [2]	**38.31**	**0.974**	**0.89**
CLSTM	29.32	0.924	2.85
3D-CNN	**35.31**	**0.964**	**1.38**

Table 2. Temporal errors on the static video test set for different methods.

5. Conclusion

In this work, we first consider the relation between consecutive reference frames using CLSTM and 3D-CNN. Quantitative and qualitative analysis demonstrate that our method achieves promising results compared with other traditional pipelines but there is still room for improvement. Besides, while 3D CNN maintains better temporal cons- istency, it often leads to blurry frames. On the other hand, CLSTM gives to better results with respect to frame quality but fails to keep temporal consistency.

Reference

[1] C. Chen, Q. Chen, J. Xu, and V. Koltun. Learning to see in the dark. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[2] C. Chen, Q. Chen, Minh N. Do, and V. Koltun. Seeing motion in the dark. In The IEEE International Conference on Computer Vision (ICCV), 2019.

[3] H. Zhang, C. Shen, Y. Li, Y. Cao, Y. Liu, and Y. Yan. Exploiting temporal consistency for real-time video depth estimation. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019