# Removal of Artifacts from Vehicle Mounted Images using Convolutional Autoencoders

Muhammad Suhail Najeeb
#1018062222

# Introduction

**Motivation:** Different weather and lighting conditions affect the performance of Image Processing tasks on Road Images.

It is crucial to enhance these images in tasks like:

- Vehicle Detection
- Traffic Sign Detection
- Autonomous Driving etc.

**Objective:** Our goal is to remove different inclement weather conditions affecting road-side images for example:

- Rain
- Snow
- Haze

# Literature Review

- Conditional Variational Image Deraining [1]
- Reconstruction of Image Sequences using a Recurrent Denoising Autoencoder [2]
- Hierarchical approach for Rain or Snow removing in Single Image [3]
- Aerial Image Dehazing using a deep Convolutional Autoencoder [4]

# Dataset

- CURE-TSD (Challenging Unreal and Real Environment for Traffic Sign Detection) [5]

The dataset contains video from 49 different sequences under different challenging conditions [6] for example:

1. Decolorization
2. Lens Blur
3. Codec Error
4. Darkening
5. Dirty Lens
6. Exposure
7. Gaussian Blur
8. Noise
9. Rain
10. Shadow
11. Haze

# Training Data

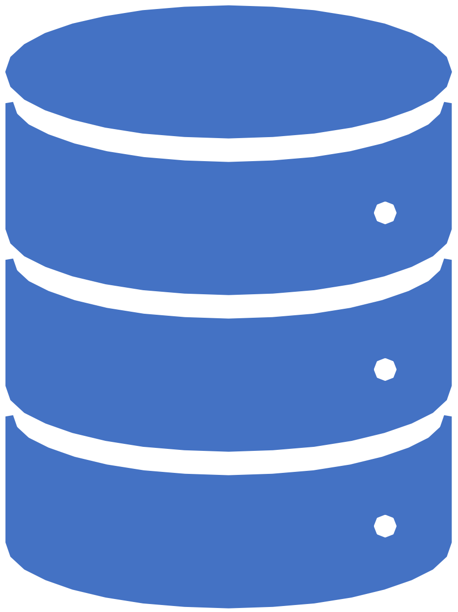
- The main dataset contains over a thousand video sequences under different challenge levels and conditions. Subsets of the original dataset were used to train the enhancement networks.

- Subsets of data selected from the original dataset is as follows:

DERAIN NETWORK – REAL DATA, CHALLENGE TYPE – RAIN, CHALLENGE LEVEL – 1

DE-SNOW NETWORK – REAL DATA, CHALLENGE TYPE – SNOW, CHALLENGE LEVEL – 2

DEHAZE NETWORK – REAL DATA, CHALLENGE TYPE – HAZE, CHALLENGE LEVEL – 2



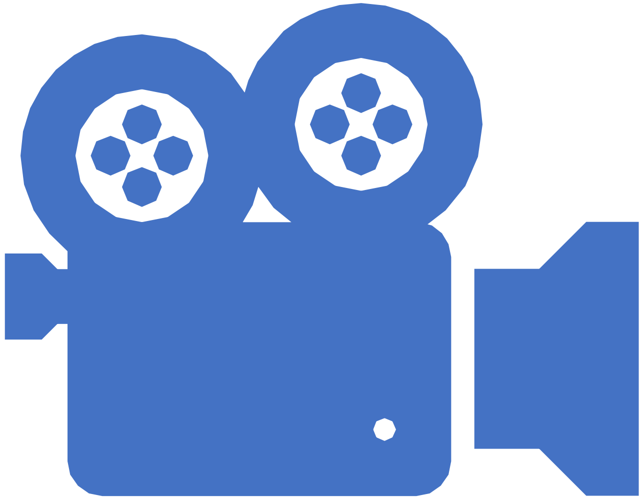
# Preprocessing

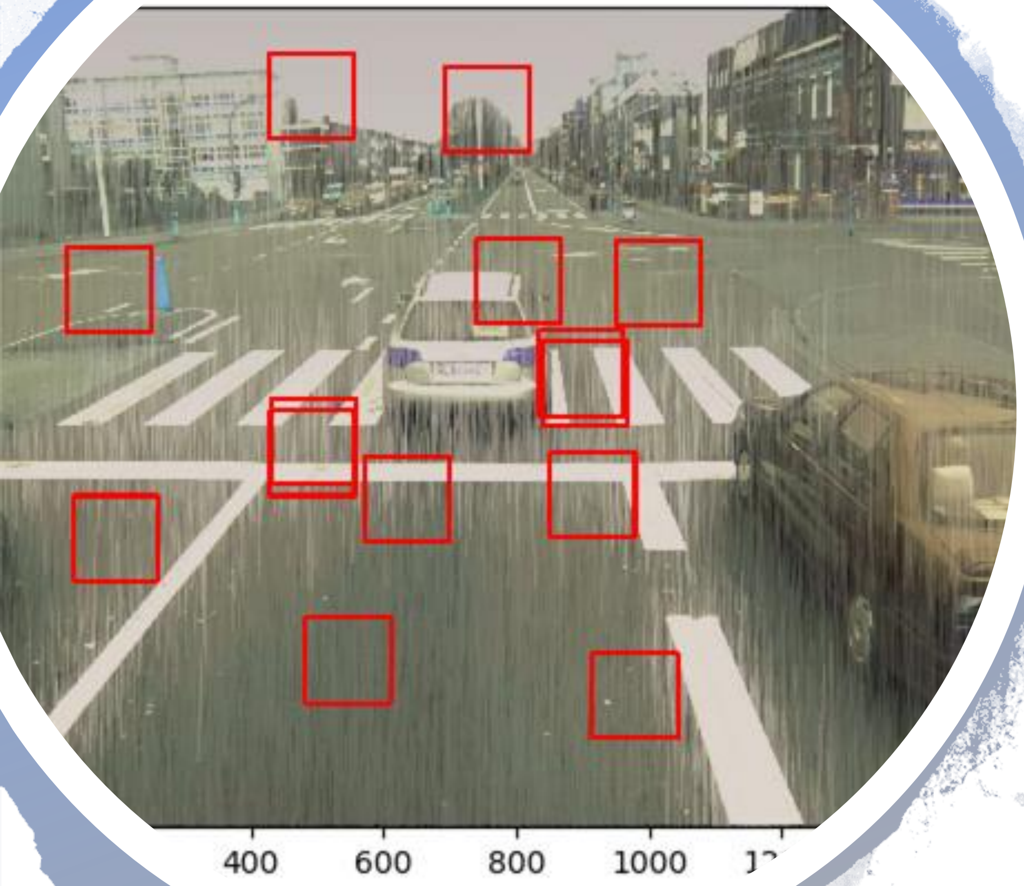
The following parts were considered for the preprocessing step:

- Frame Extraction
- Cropping of Patches
- Dataset Creation

# Frame Extraction

- Each video sequence:
  - Resolution: 1628 x 1236
  - Sequence Length: 30 seconds
  - Frame rate: 10 fps
- Each Video sequence generated 300 frames.
- Total No. of frames generated per challenge:
  - 300*49 = 14700





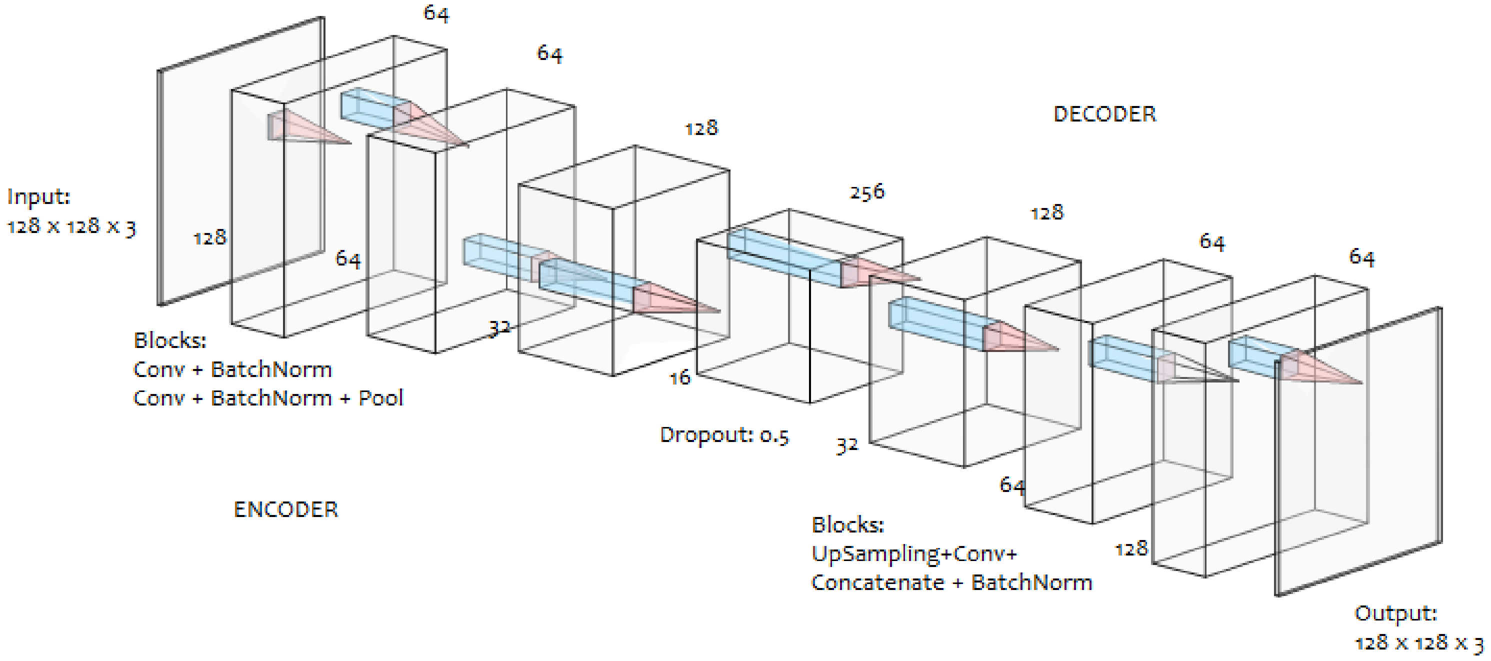

# Cropping of Patches

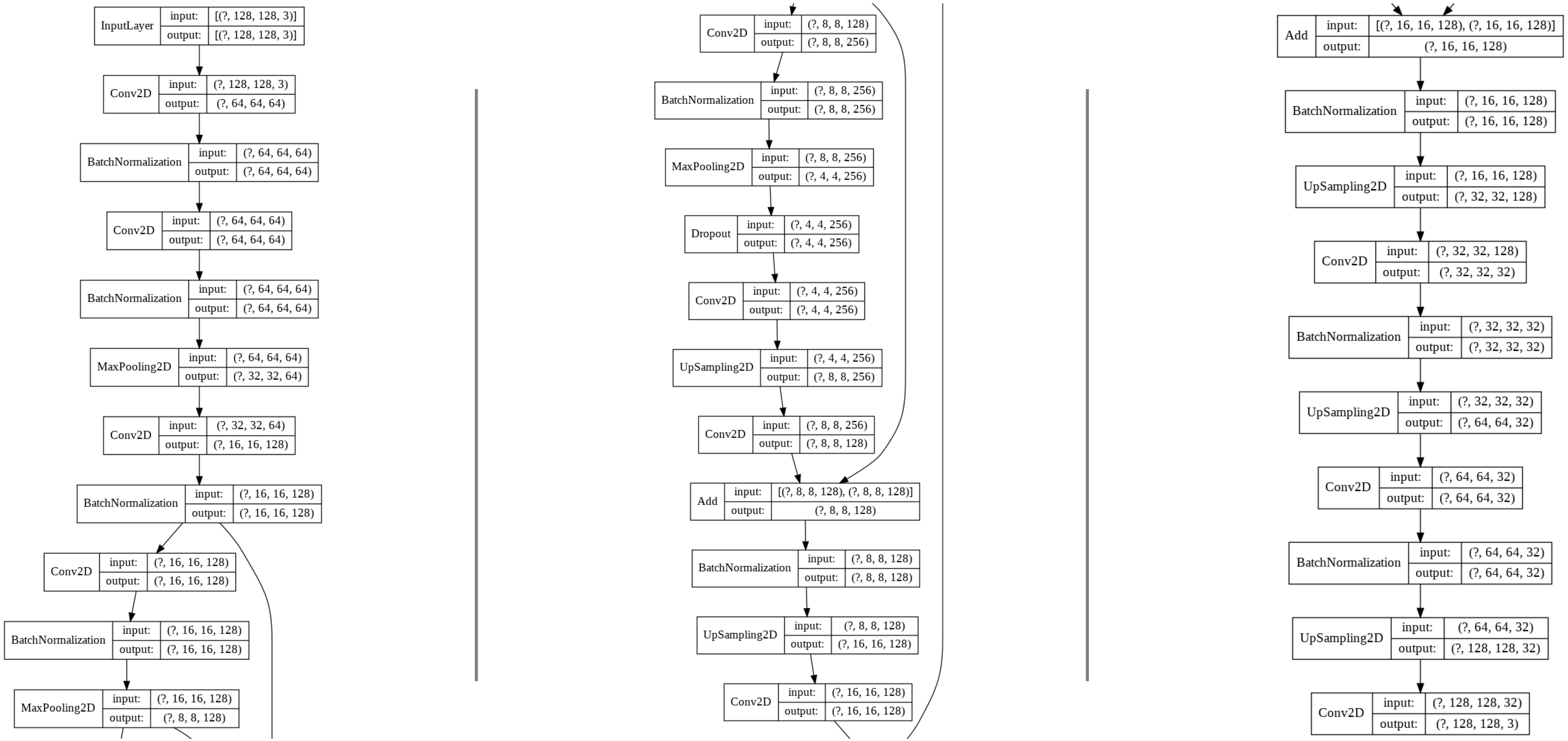
- Instead of enhancing the whole image at once, we aim at cropping random patches from the image and separately enhance them.
- The patch is 128 x 128 pixels.
- 8 Random patches are taken from each extracted video frame.

# Dataset Creation

- 8 Random Patches are taken from each frame extracted from the video sequences.
- 49 Video sequences with 300 frames each,
- Total number of random patches for training: 49*8*300 = 1,176,000
- The patches are stored in hdf5 data files for faster training
- Each data file is around 6 GB.

# Autoencoder Architecture



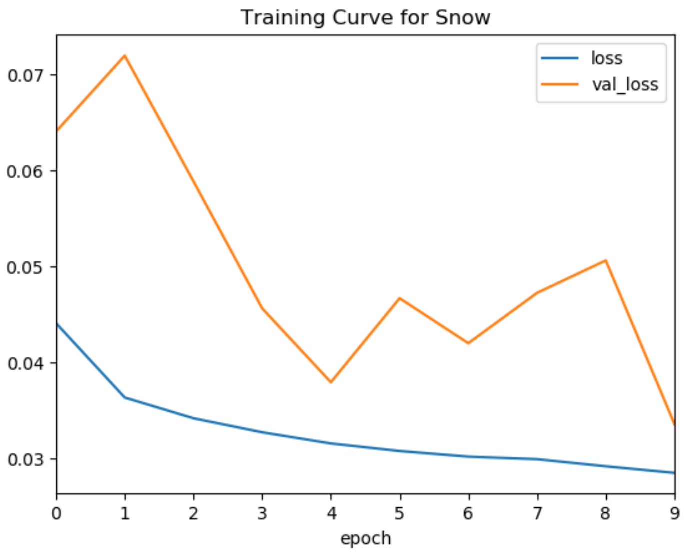


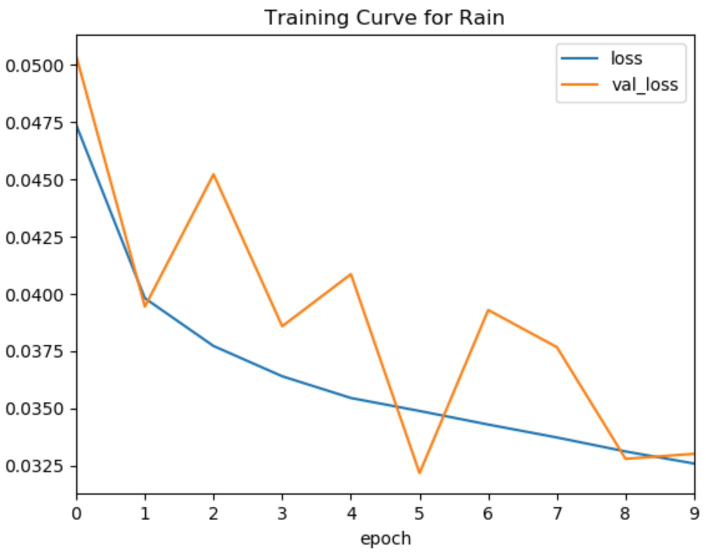
InputLayer input: [(?, 128, 128, 3)] output: [(?, 128, 128, 3)]
Conv2D input: (?, 128, 128, 3) output: (?, 64, 64, 64)
BatchNormalization input: (?, 64, 64, 64) output: (?, 64, 64, 64)
Conv2D input: (?, 64, 64, 64) output: (?, 64, 64, 64)
BatchNormalization input: (?, 64, 64, 64) output: (?, 64, 64, 64)
MaxPooling2D input: (?, 64, 64, 64) output: (?, 32, 32, 64)
Conv2D input: (?, 32, 32, 64) output: (?, 16, 16, 128)
BatchNormalization input: (?, 16, 16, 128) output: (?, 16, 16, 128)
Conv2D input: (?, 16, 16, 128) output: (?, 16, 16, 128)
BatchNormalization input: (?, 16, 16, 128) output: (?, 16, 16, 128)
MaxPooling2D input: (?, 16, 16, 128) output: (?, 8, 8, 128)
Conv2D input: (?, 8, 8, 128) output: (?, 8, 8, 256)
BatchNormalization input: (?, 8, 8, 256) output: (?, 8, 8, 256)
MaxPooling2D input: (?, 8, 8, 256) output: (?, 4, 4, 256)
Dropout input: (?, 4, 4, 256) output: (?, 4, 4, 256)
Conv2D input: (?, 4, 4, 256) output: (?, 4, 4, 256)
UpSampling2D input: (?, 4, 4, 256) output: (?, 8, 8, 256)
Conv2D input: (?, 8, 8, 256) output: (?, 8, 8, 128)
Add input: [(?, 8, 8, 128), (?, 8, 8, 128)] output: (?, 8, 8, 128)
BatchNormalization input: (?, 8, 8, 128) output: (?, 8, 8, 128)
UpSampling2D input: (?, 8, 8, 128) output: (?, 16, 16, 128)
Conv2D input: (?, 16, 16, 128) output: (?, 16, 16, 128)
Add input: [(?, 16, 16, 128), (?, 16, 16, 128)] output: (?, 16, 16, 128)
BatchNormalization input: (?, 16, 16, 128) output: (?, 16, 16, 128)
UpSampling2D input: (?, 16, 16, 128) output: (?, 32, 32, 128)
Conv2D input: (?, 32, 32, 128) output: (?, 32, 32, 32)
BatchNormalization input: (?, 32, 32, 32) output: (?, 32, 32, 32)
UpSampling2D input: (?, 32, 32, 32) output: (?, 64, 64, 32)
Conv2D input: (?, 64, 64, 32) output: (?, 64, 64, 32)
BatchNormalization input: (?, 64, 64, 32) output: (?, 64, 64, 32)
UpSampling2D input: (?, 64, 64, 32) output: (?, 128, 128, 32)
Conv2D input: (?, 128, 128, 32) output: (?, 128, 128, 3)

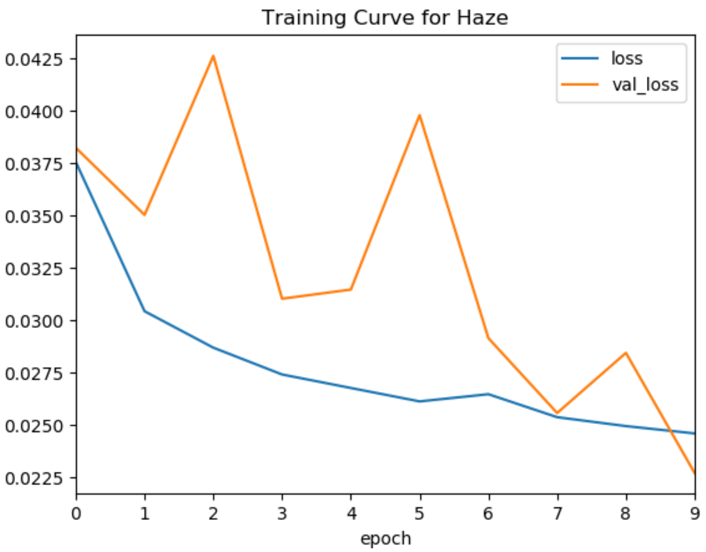
# Training

- Framework: TensorFlow/Keras
- Loss: L1 Loss (Mean Absolute Error)
- Metrics: SSIM (Structural Similarity), PSNR (Peak Signal-to-Noise Ratio)
- Optimizer: Adam
- Epochs = 10

# Training Curves



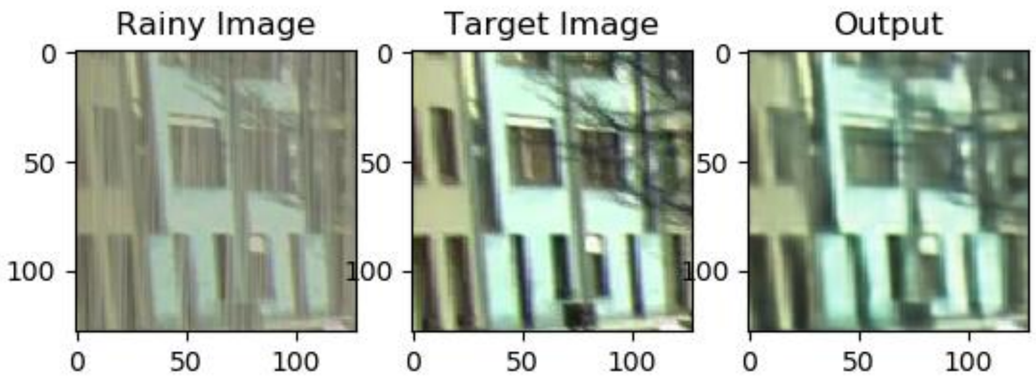



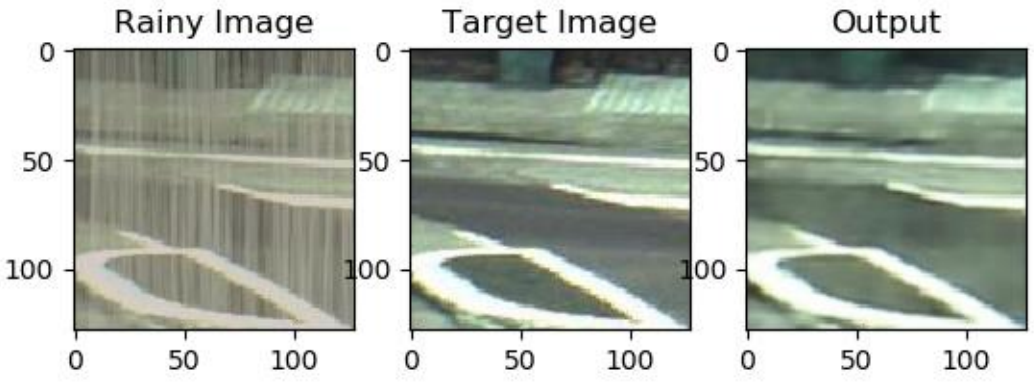



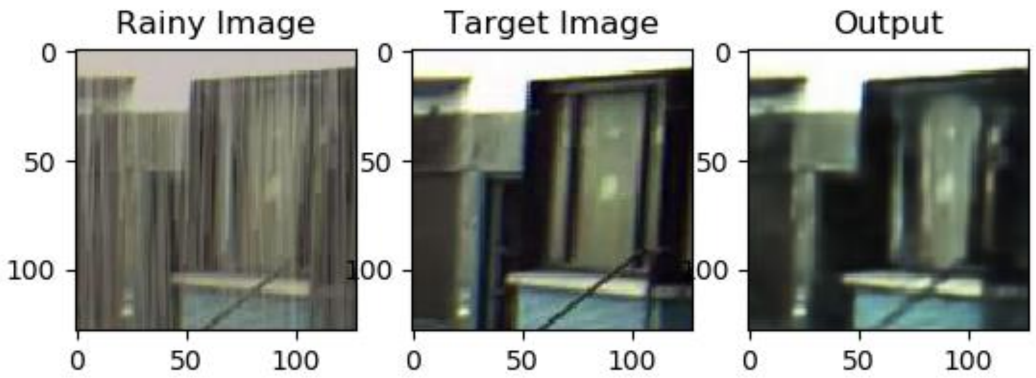
# Training Results (Rain)

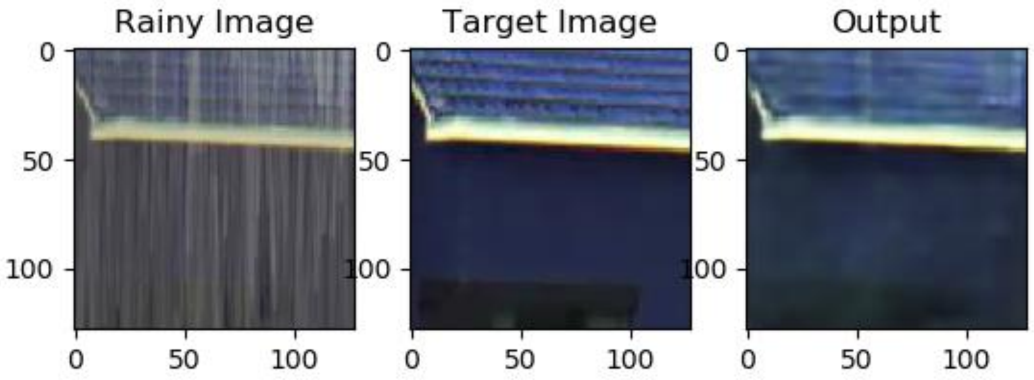
| epoch | Loss(MAE) | PSNR (dB) (train) | SSIM (train) | Loss(MAE) (validation) | PSNR (dB) (validation) | SSIM (validation) |
|---|---|---|---|---|---|---|
| 0 | 0.047374 | 24.77054 | 0.817058 | 0.050384 | 24.96131 | 0.836987 |
| 1 | 0.039813 | 26.18734 | 0.852156 | 0.039447 | 26.3275 | 0.857574 |
| 2 | 0.037729 | 26.65196 | 0.863964 | 0.045219 | 25.46388 | 0.864368 |
| 3 | 0.036408 | 26.95132 | 0.870426 | 0.038591 | 26.24665 | 0.871962 |
| 4 | 0.035461 | 27.16986 | 0.874645 | 0.040861 | 26.03734 | 0.871685 |
| 5 | 0.034895 | 27.30202 | 0.877449 | 0.032185 | 27.61251 | 0.879807 |
| 6 | 0.034303 | 27.44192 | 0.880037 | 0.039299 | 25.83583 | 0.877998 |
| 7 | 0.033738 | 27.57442 | 0.882197 | 0.037678 | 26.73658 | 0.883709 |
| 8 | 0.033128 | 27.71549 | 0.884198 | 0.032807 | 27.95652 | 0.886174 |
| 9 | 0.032601 | 27.83729 | 0.885687 | 0.033027 | 27.69954 | 0.887347 |

# Sample Images (Rain)



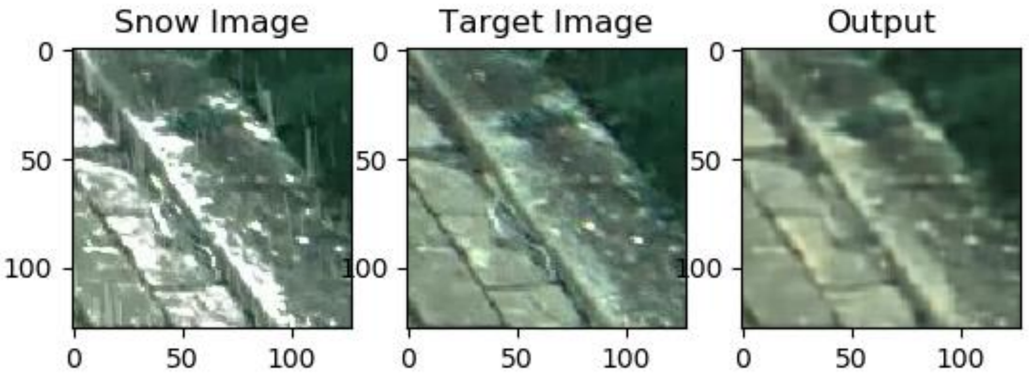



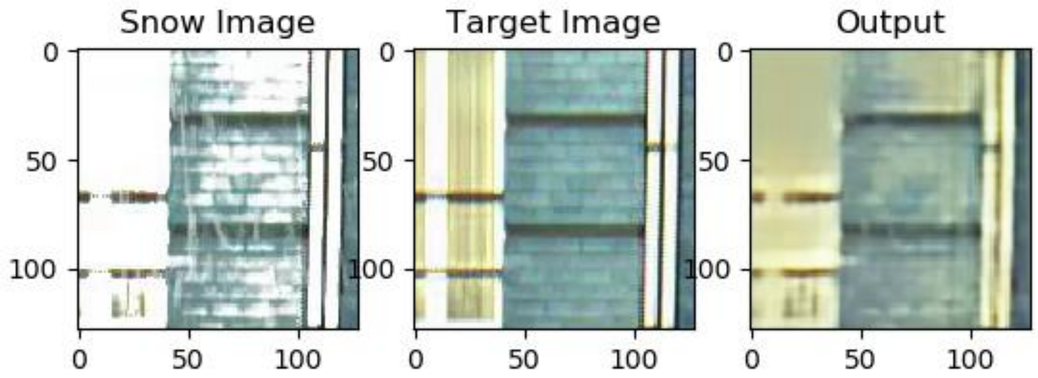



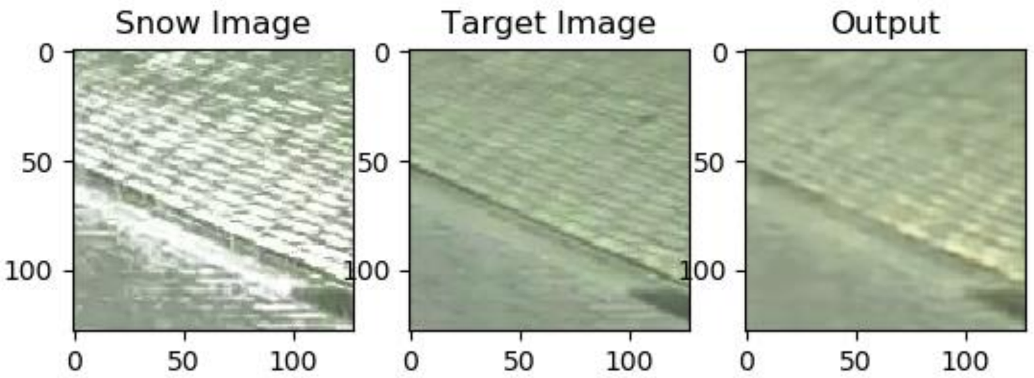
# Training Results (Snow)

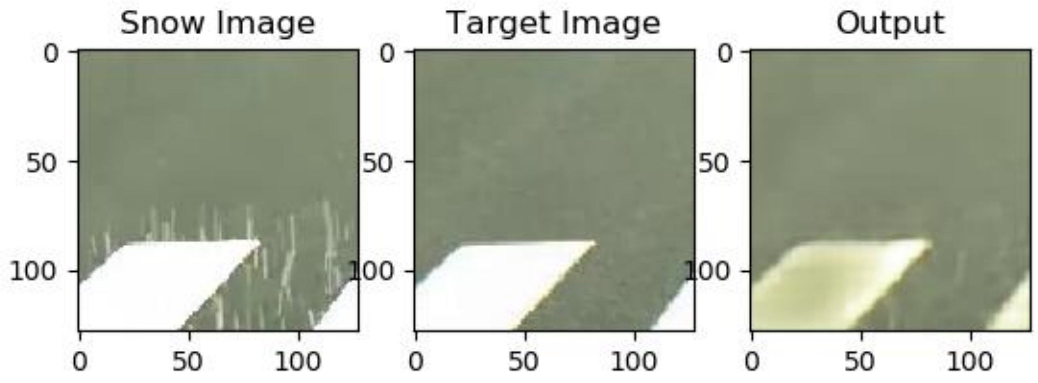
| epoch | Loss(MAE) | PSNR (dB) (train) | SSIM (train) | Loss(MAE) (validation) | PSNR (dB) (validation) | SSIM (validation) |
|---|---|---|---|---|---|---|
| 0 | 0.044121 | 25.38874 | 0.832442 | 0.064059 | 23.93991 | 0.835107 |
| 1 | 0.03638 | 26.94942 | 0.867814 | 0.071959 | 23.00139 | 0.854152 |
| 2 | 0.034222 | 27.4757 | 0.879834 | 0.058885 | 25.02603 | 0.86158 |
| 3 | 0.032764 | 27.83519 | 0.88647 | 0.045662 | 25.91427 | 0.874287 |
| 4 | 0.0316 | 28.12929 | 0.891145 | 0.037969 | 26.92684 | 0.889801 |
| 5 | 0.030814 | 28.33237 | 0.894522 | 0.046713 | 26.46678 | 0.883507 |
| 6 | 0.030231 | 28.48101 | 0.896852 | 0.042033 | 26.61473 | 0.889001 |
| 7 | 0.029963 | 28.54774 | 0.897515 | 0.047273 | 25.53843 | 0.887453 |
| 8 | 0.029223 | 28.74194 | 0.900511 | 0.050649 | 26.99825 | 0.887477 |
| 9 | 0.02854 | 28.9308 | 0.902583 | 0.033612 | 27.4818 | 0.897646 |

# Sample Images (Snow)



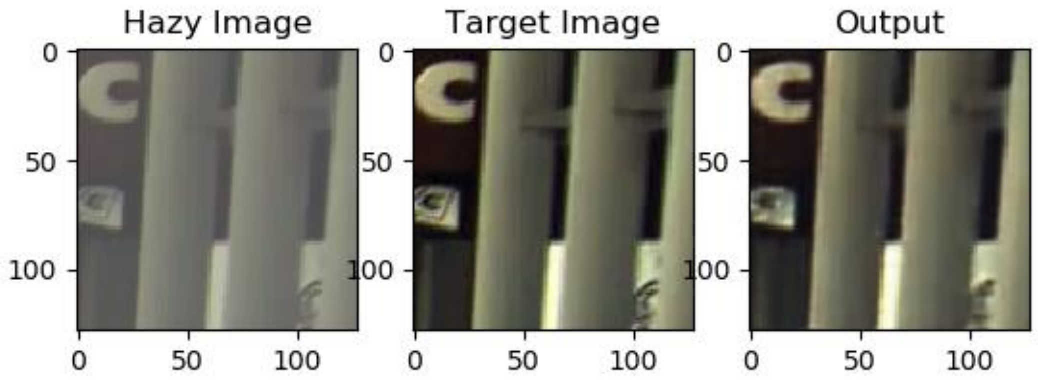



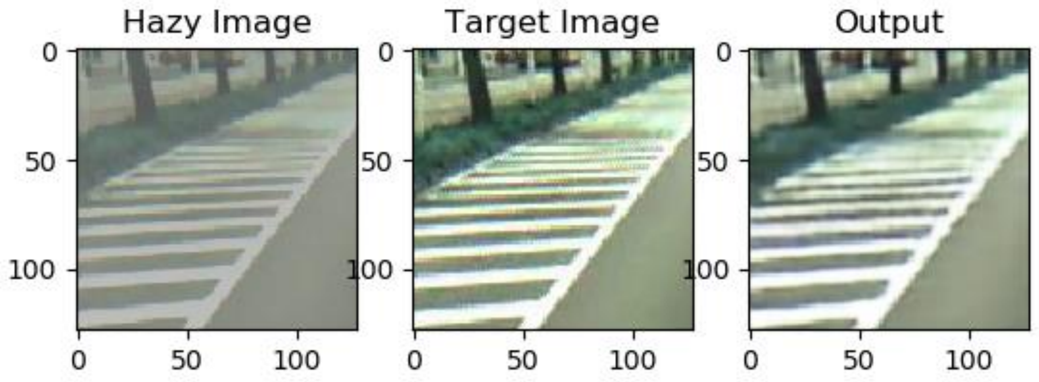



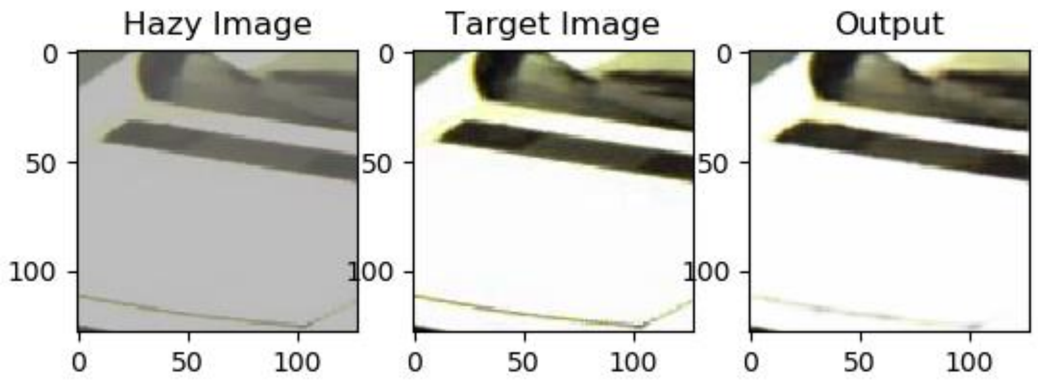
# Training Results (Haze)

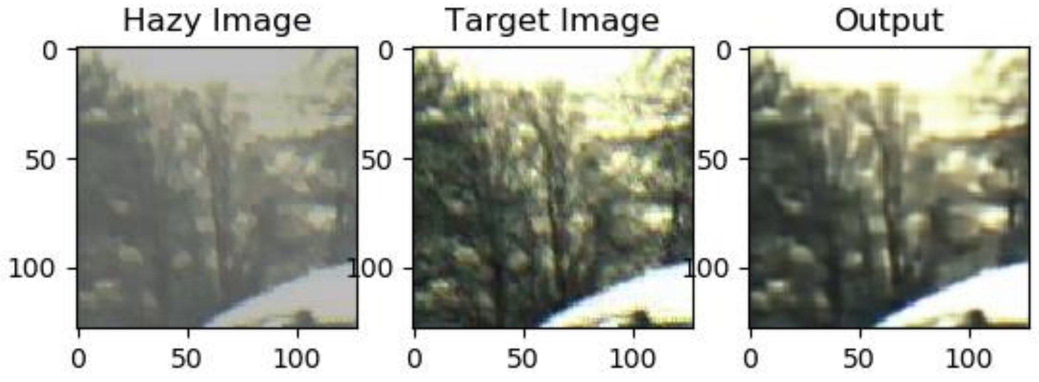
| epoch | Loss(MAE) | PSNR (dB) (train) | SSIM (train) | Loss(MAE) (validation) | PSNR (dB) (validation) | SSIM (validation) |
|---|---|---|---|---|---|---|
| 0 | 0.037567 | 26.52969 | 0.844182 | 0.038249 | 26.54979 | 0.861328 |
| 1 | 0.030438 | 28.29114 | 0.881497 | 0.035039 | 27.32708 | 0.883204 |
| 2 | 0.028697 | 28.85748 | 0.895195 | 0.042638 | 26.1263 | 0.888372 |
| 3 | 0.027415 | 29.28261 | 0.903512 | 0.031036 | 28.38426 | 0.898749 |
| 4 | 0.02677 | 29.49814 | 0.907474 | 0.031472 | 27.99859 | 0.902625 |
| 5 | 0.026129 | 29.7161 | 0.910647 | 0.039807 | 26.36491 | 0.905225 |
| 6 | 0.026473 | 29.60702 | 0.909585 | 0.029152 | 28.85308 | 0.911686 |
| 7 | 0.025375 | 29.97459 | 0.914358 | 0.025575 | 29.64278 | 0.916766 |
| 8 | 0.02495 | 30.11279 | 0.915859 | 0.028448 | 28.83317 | 0.913907 |
| 9 | 0.024598 | 30.23061 | 0.917335 | 0.022705 | 30.46076 | 0.917678 |

# Sample Images (Haze)



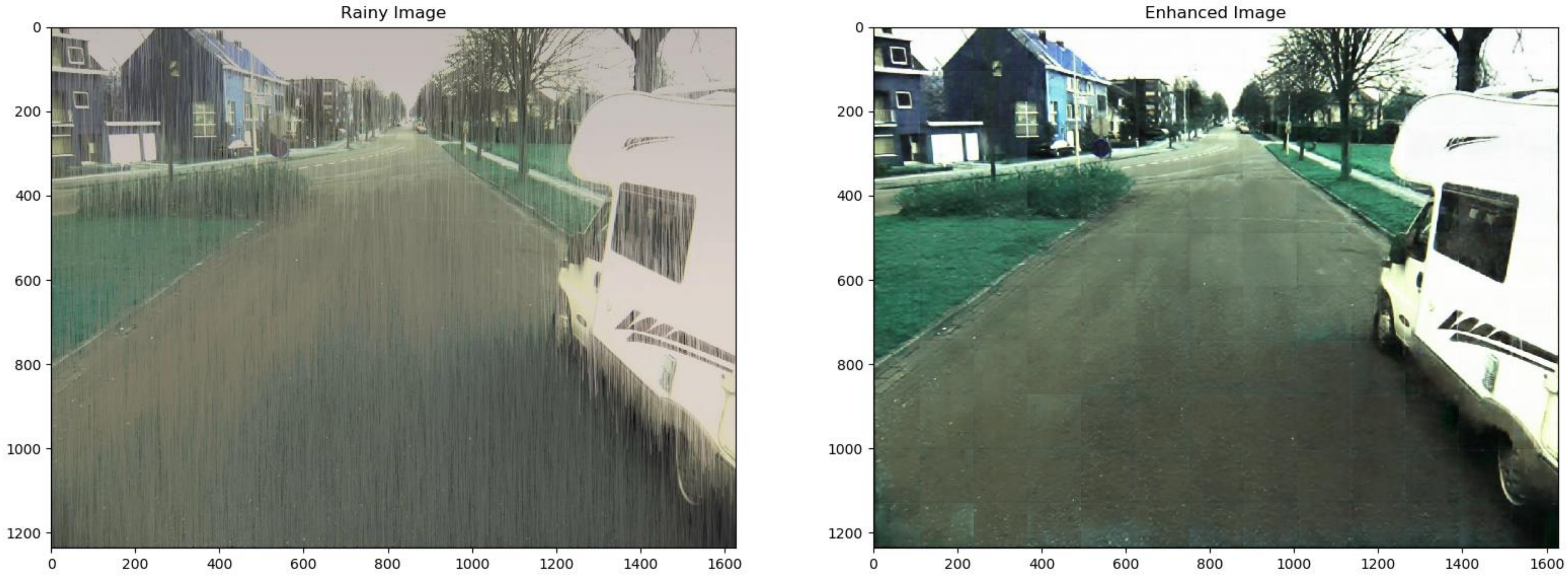



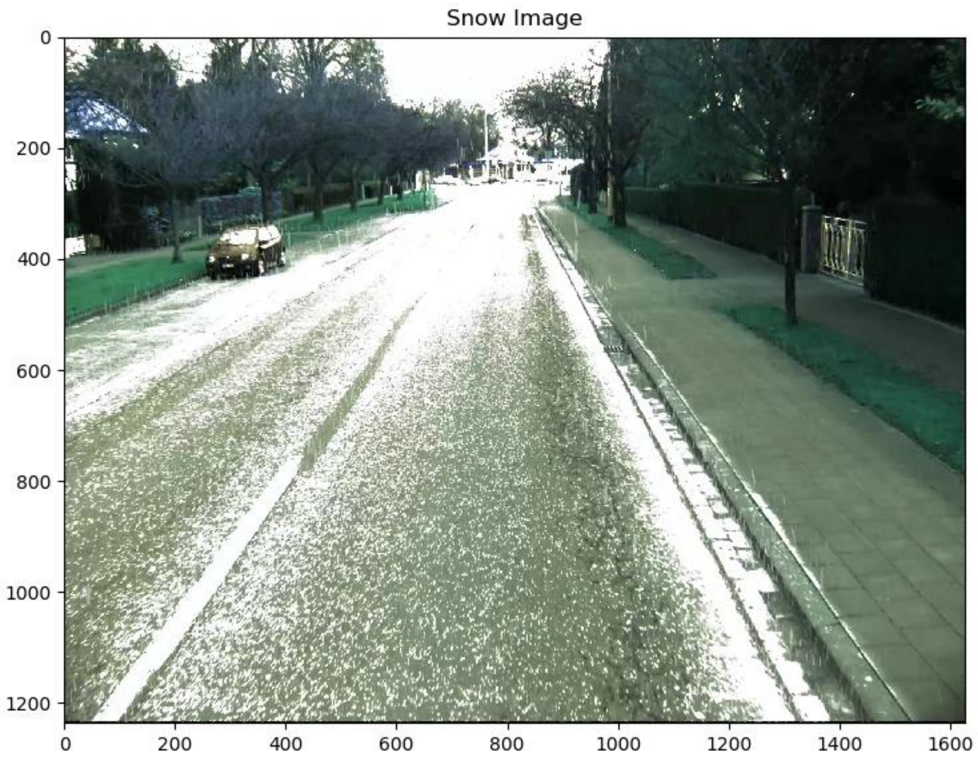
# Stitching

- Full size frames have a resolution of 1628 x 1236 pixels
- The full frame is split into multiple patches of size 128 x 128 pixels
- The patches are passed through the network to enhance the image
- Enhanced patches are stitched together to produce final image.
- Drawback: Checkerboard pattern at output, can be postprocessed and removed using filters

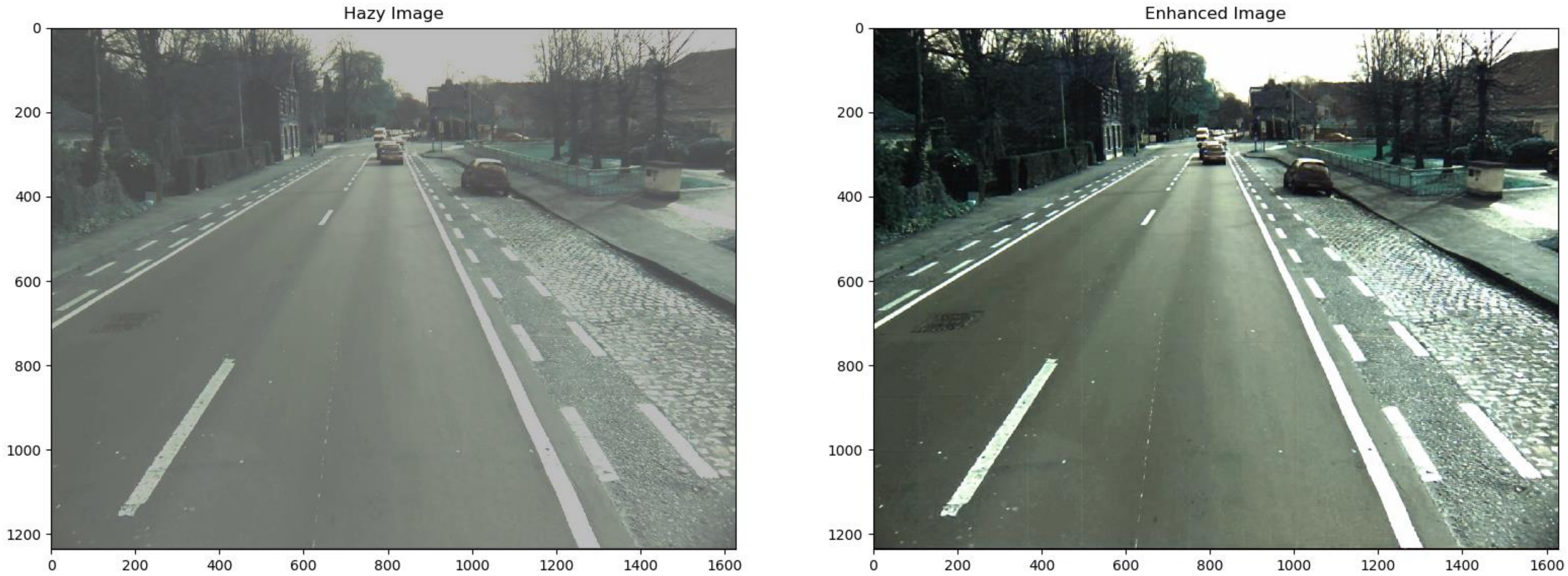
# Full Frame Results



# Full Frame Results




# Full Frame Results



# Conclusion and Future Scope

- The simple model already performs convincingly to remove Rain, Snow & Haze
- This might have real-world applications to improve the performance of computer vision techniques with vehicles.
- The model might perform well on other artifacts as well such as dirty lens, exposure correction, noise, codec error etc.
- The effect of adding skip connections, recurrent blocks and/or residual blocks to the architecture might be studied
- Incorporating time-domain information might greatly improve performance (i.e. we can take more than 1 frame for prediction)

# References


[1] Du Y, Xu J, Zhen X, Cheng MM, Shao L. Conditional Variational Image Deraining. IEEE Transactions on Image Processing. 2020 May.

[2] Chaitanya CR, Kaplanyan AS, Schied C, Salvi M, Lefohn A, Nowrouzezahrai D, Aila T. Interactive reconstruction of Monte Carlo image sequences using a recurrent denoising autoencoder. ACM Transactions on Graphics (TOG). 2017 Jul 20;36(4):1-2.

[3] Wang Y, Liu S, Chen C, Zeng B. A hierarchical approach for rain or snow removing in a single color image. IEEE Transactions on Image Processing. 2017 May 26;26(8):3936-50.

[4] Fazlali H, Shirani S, McDonald M, Brown D, Kirubarajan T. Aerial image dehazing using a deep convolutional autoencoder. Multimedia Tools and Applications. 2020 Oct;79(39):29493-511.

# References

[5] Temel D, Chen MH, AlRegib G. Traffic sign detection under challenging conditions: A deeper look into performance variations and spectral characteristics. IEEE Transactions on Intelligent Transportation Systems. 2019 Aug 9.

[6] Temel D, Alshawi T, Chen MH, AlRegib G. Challenging environments for traffic sign detection: Reliability assessment under inclement conditions. arXiv preprint arXiv:1902.06857. 2019 Feb 19.

# Thank You!

For Source Code: https://github.com/suhailnajeeb/cure-tsd-revisit/

Questions? suhailnajeeb19@gmail.com