Image Denoising and Deblurring

Srivatsan Sarvesan, Tarang Mendhe
IIT Madras

November 6, 2024

1 Introduction

In recent years, the field of image restoration has garnered significant attention due to its critical applications in various domains, including medical imaging, surveillance, and industrial inspection. The challenge of recovering clean images from noisy and blurred inputs while preserving important features—particularly defects of interest—has motivated the development of advanced neural network architectures. In this work, we propose a novel image restoration network built upon the well-established U-Net framework, enhanced with spatial and channel attention (SCA) mechanisms and non-local blocks. This architecture is designed to effectively capture both local and global dependencies within images, addressing the dual objectives of noise and blur removal while maintaining the integrity of subtle features that are essential for accurate defect identification. By leveraging attention mechanisms, our network can dynamically prioritize significant areas during the restoration process, thus ensuring that critical defects are preserved in the output. The following sections detail the methodology employed in our proposed network, which integrates innovative architectural elements to enhance image restoration performance.

2 Methodology

2.1 Network Overview

The proposed network, Figure 1 is designed for **Image Restoration** to recover clean images from **noisy and blurred inputs** while **preserving defects of interest**. The network architecture leverages a **U-Net framework** [1] augmented with **spatial and channel attention (SCA) mechanisms** [2] and **non-local blocks** [3] to efficiently capture local and global dependencies in images. These enhancements address the dual task of removing noise and blur while retaining critical regions, specifically defects of interest, which are often challenging to preserve due to their subtle features.

In this architecture, the **encoder** progressively reduces spatial dimensions, extracting essential features from various resolutions, while the **decoder** uses

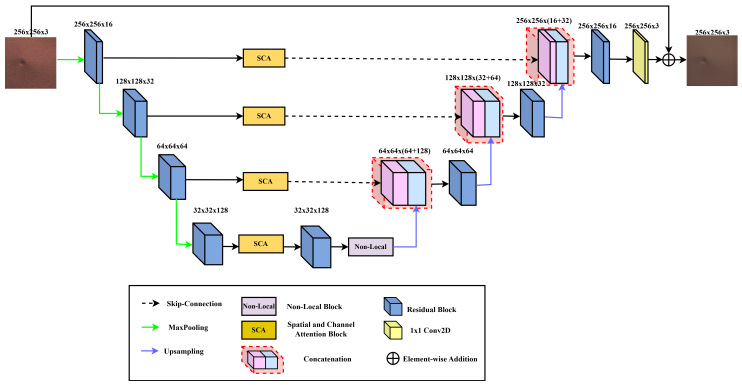


Figure 1: Architecture of the proposed network for image restoration.

these features to reconstruct the output image, with skip connections allowing for the seamless transfer of high-resolution details between corresponding encoder and decoder layers. The **attention blocks** embedded in skip connections and bottleneck layers adaptively highlight significant areas, helping the network prioritize defects while suppressing irrelevant noise and blur. Additionally, a **non-local block** is introduced in the bottleneck, which enhances the global perception capability, crucial for capturing long-range dependencies and achieving robust restoration.

2.2 Encoder

The encoder aims to capture **multi-scale feature representations** from the noisy, blurred input image. It consists of multiple **double convolution** blocks, each followed by **max-pooling** layers, which progressively downsample the spatial dimensions while increasing the number of feature channels. This structure allows the network to gather spatial information effectively while increasing the receptive field.

Each double convolution block comprises two 3x3 convolution layers followed by a ReLU activation function. These layers help capture finer details and higher-order spatial features, which are crucial for detecting defects. Additionally, **residual connections** [4] within each double convolution block enable efficient gradient propagation and help mitigate vanishing gradient issues during training.

The forward pass for each double convolution block in the encoder can be mathematically represented as:

$$\text{ConvBlock}_i(\mathbf{X}) = ReLU\left(\mathbf{W}_{i2} * \left(ReLU\left(\mathbf{W}_{i1} * \mathbf{X} + \mathbf{b}_{i1}\right)\right) + \mathbf{b}_{i2}\right) \tag{1}$$

where $\mathbf{W}_{i1}$ and $\mathbf{W}_{i2}$ are the weights of the two convolution layers, $\mathbf{b}_{i1}$ and $\mathbf{b}_{i2}$ are biases, $*$ denotes convolution, and (ReLU) represents the ReLU activation.

2.3 Decoder

The decoder reconstructs the restored image from encoded features, incorporating information from both coarse and fine levels using **upsampling** operations and **skip connections**. Each upsampling step doubles the spatial resolution using **bilinear interpolation** and concatenates it with the corresponding encoder features via skip connections, ensuring that high-resolution details are preserved throughout the restoration process.

Following each upsampling step, a **double convolution block** processes the concatenated features to improve feature integration, enhancing the network's ability to generate high-quality reconstructions.

The decoding process can be formulated as:

$$\mathbf{Y}_i = \text{ConvBlock}_i\left(\text{Upsample}(\mathbf{X}_{i+1}) \oplus \mathbf{F}_{\text{skip},i}\right) \tag{2}$$

where $\mathbf{X}_{i+1}$ represents the upsampled feature map from the deeper layer, $\mathbf{F}_{\text{skip},i}$ denotes the corresponding encoder features, and $\oplus$ represents the concatenation operation.

2.4 Attention Mechanisms

Attention mechanisms play a vital role in highlighting critical regions—particularly defects of interest—by allowing the network to weigh features based on their relevance to the restoration task. This work incorporates a **Spatial and Channel Attention (SCA)** block and a **Non-Local Block** to enhance both spatial and global feature dependencies.

2.4.1 Spatial and Channel Attention (SCA) Block

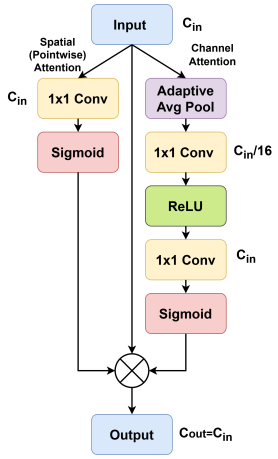


Figure 2: Spatial and Channel Attention (SCA) Block

The **SCA block** as shown in figure 2 comprises separate **spatial** and **channel attention modules**. The spatial attention module utilizes pointwise convolution to highlight spatial locations critical for defect preservation. In contrast, the channel attention module, based on global average pooling, refines the feature maps along the channel dimension.

Spatial Attention A pointwise convolution layer generates a spatial attention map, indicating the importance of each pixel location. This can be formulated as:

$$\text{SpatialAttention}(\mathbf{X}) = \sigma(\mathbf{W}_{\text{spatial}} * \mathbf{X} + \mathbf{b}_{\text{spatial}}) \tag{3}$$

where $\mathbf{W}_{\text{spatial}}$ and $\mathbf{b}_{\text{spatial}}$ are the weights and bias for spatial attention, and σ denotes the sigmoid activation function.

Channel Attention The channel attention mechanism captures the interdependencies between feature maps. The process begins with global average pooling, followed by a fully connected layer and activation functions:

$$\text{ChannelAttention}(\mathbf{X}) = \sigma\left(\mathbf{W}_{\text{ch2}}\,\text{ReLU}(\mathbf{W}_{\text{ch1}}\,\text{GAP}(\mathbf{X}) + \mathbf{b}_{\text{ch1}}) + \mathbf{b}_{\text{ch2}}\right) \tag{4}$$

where GAP represents global average pooling, $\mathbf{W}_{\text{ch1}}$ and $\mathbf{W}_{\text{ch2}}$ are the weights of fully connected layers, and σ is the sigmoid function.

The final SCA output is computed as:

$$\mathbf{X}_{\text{SCA}} = \mathbf{X} \times \text{SpatialAttention}(\mathbf{X}) \times \text{ChannelAttention}(\mathbf{X}) \tag{5}$$

where $\times$ denotes element-wise multiplication, which integrates the attention-modulated spatial and channel features.

2.4.2 Non-Local Block

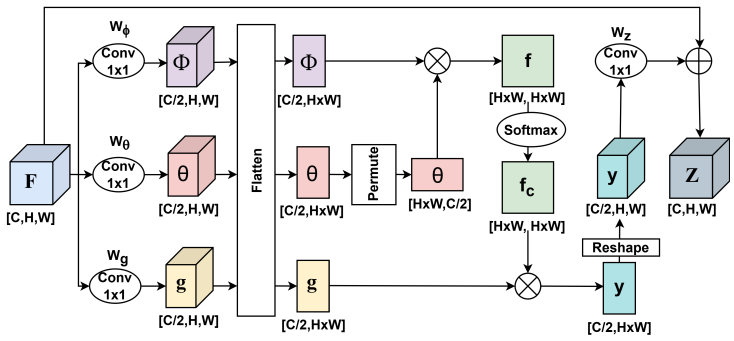


Figure 3: Non-Local Block

The **Non-Local Block** [3] shown in figure 3, is employed in the bottleneck layer to capture global dependencies by relating each pixel to every other pixel in

the feature map. This module enables the model to gather contextual information from distant spatial locations, which is especially important for preserving fine defects in the image.

The non-local operation can be represented as:

$$\mathbf{Y} = \mathbf{X} + \mathbf{W}\left(\text{Softmax}\left(\theta(\mathbf{X})^T \cdot \phi(\mathbf{X})\right) \cdot \mathbf{g}(\mathbf{X})\right) \tag{6}$$

where $\theta(\mathbf{X}) = \mathbf{W}_\theta * \mathbf{X}$, $\phi(\mathbf{X}) = \mathbf{W}_\phi * \mathbf{X}$, and $\mathbf{g}(\mathbf{X}) = \mathbf{W}_g * \mathbf{X}$ are learned transformations of the input feature map. The resulting attention map is computed by matrix-multiplying $\theta(\mathbf{X})$ and $\phi(\mathbf{X})$, and then normalized using a softmax function. The attended feature map $\mathbf{g}(\mathbf{X})$ is then projected back to the original feature dimensions through $\mathbf{W}$.

This operation introduces a residual connection by adding $\mathbf{Y}$ back to $\mathbf{X}$, improving gradient flow and helping the network learn robust global features for image restoration.

3 Experimental Settings

3.1 Dataset Description

The MVTec Anomaly Detection (AD) dataset, a benchmark for image anomaly detection tasks, was used in this study. For the purpose of image restoration from noisy and blurred images, the images in this dataset were artificially degraded by adding noise and blur, thereby creating "degraded" images. The model takes these degraded images as input and generates restored images that closely resemble the clean, original images. All images were resized to a resolution of 256×256 pixels. The dataset comprises:

- **Training Set:** 932 degraded images with corresponding clean images as ground truth.
- **Validation Set:** 88 degraded images with corresponding clean images as ground truth.

Additionally, defect masks are provided to ensure that critical regions with anomalies are preserved in the restored images. The task is focused on accurately restoring image quality while preserving areas of interest.

3.2 Experiments and Analysis

The experiments were conducted using a proposed U-Net model with a total of 867,842 parameters, implemented in PyTorch. The model was trained for 250 epochs on an NVIDIA T4 GPU using the Adam optimizer with a learning rate of 0.001. During training, a composite loss function was used, which combines the following loss functions:

- **Charbonnier Loss** $\mathcal{L}_{\text{char}}$, which is a smooth approximation of the $L1$ loss:

$$\mathcal{L}_{\text{char}} = \frac{1}{N} \sum_{i=1}^{N} \sqrt{\|x_i - y_i\|^2 + \epsilon^2},$$

  where N denotes the total number of pixels in the image, where x_i represents the predicted pixel value at position i, and y_i denotes the ground truth pixel value at position i. The term ϵ is a small constant, set to 10^{-3} in our experiment, which serves to ensure numerical stability in the computation of the loss.

- **SSIM Loss** $\mathcal{L}_{\text{ssim}}$, based on the Structural Similarity Index Measure (SSIM) to preserve perceptual quality:

$$\mathcal{L}_{\text{ssim}} = 1 - \text{SSIM}(x, y).$$

- **Edge Loss** $\mathcal{L}_{\text{edge}}$ captures differences in edge structures by computing the Charbonnier Loss between the Laplacian-filtered predicted and ground truth images:

$$\mathcal{L}_{\text{edge}} = \sqrt{\|\Delta(X) - \Delta(Y)\|^2 + \epsilon^2},$$

  where Δ denotes the Laplacian operator, X is the predicted image, Y is the ground truth image, and ϵ is set to 10^{-3} .

The overall loss function used for training is a weighted combination:

$$\mathcal{L} = \mathcal{L}_{\text{char}} + 0.1 \cdot \mathcal{L}_{\text{edge}} + 0.3 \cdot \mathcal{L}_{\text{ssim}}.$$

The model's performance was monitored on the validation set after each epoch. The model achieving the highest SSIM score on the validation set was saved for further evaluation.

3.3 Evaluation Metrics

To quantitatively assess the model's performance, we used the following metrics:

- **Peak Signal-to-Noise Ratio (PSNR):** Measures the ratio between the maximum possible power of a signal and the power of noise that affects the quality of its representation. PSNR is defined as:

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{\text{MAX}^2}{\text{MSE}} \right),$$

  where MAX is the maximum pixel intensity value, and MSE is the Mean Squared Error between the restored and ground truth images.

- **Structural Similarity Index Measure (SSIM):** Evaluates the structural similarity between two images, with values ranging from -1 to 1 (1 being perfect similarity). SSIM is calculated as:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)},$$

  where μ_x and μ_y are the mean intensities, σ_x and σ_y are the variances, and σ_{xy} is the covariance of images x and y. Constants C_1 and C_2 stabilize the division.

The metrics PSNR and SSIM provide insight into the perceptual quality and accuracy of the restored images relative to the ground truth images. Higher PSNR and SSIM values indicate better performance.

4 Validation Set PSNR/SSIM Results

This document presents the PSNR and SSIM values for each object in the validation set. The bar charts below show the PSNR and SSIM values for each object type, arranged alphabetically (e.g., `bottle` first, `cable` second). The average PSNR and SSIM values for the entire validation set are also reported.

Average PSNR and SSIM for Entire Validation Set

The bar charts in Figures 4 and 5 illustrate the PSNR and SSIM values for each object type, respectively. The average values for the entire validation set are reported above, indicating the general performance of the image restoration model across different object types.

- **Average PSNR:** 30.55 dB
- **Average SSIM:** 0.8450

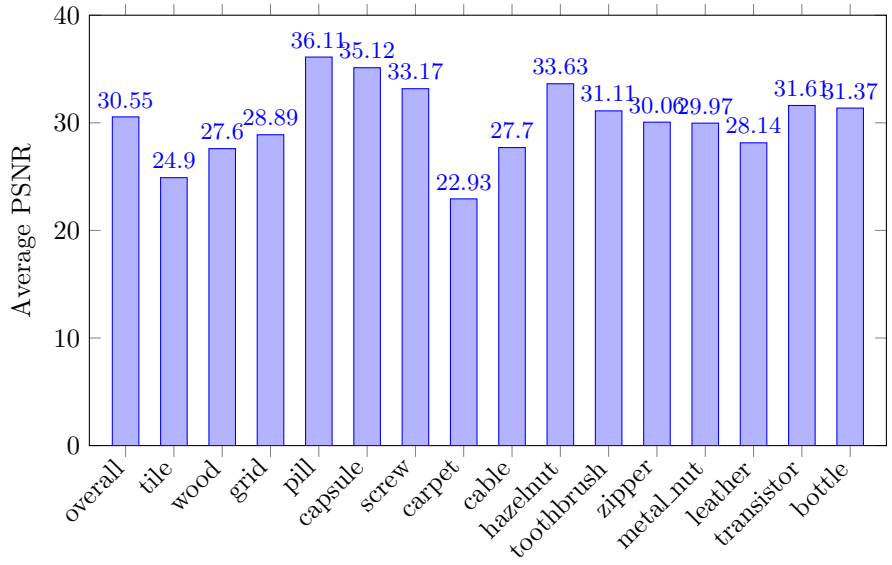


Figure 4: Average PSNR Values for Each Category

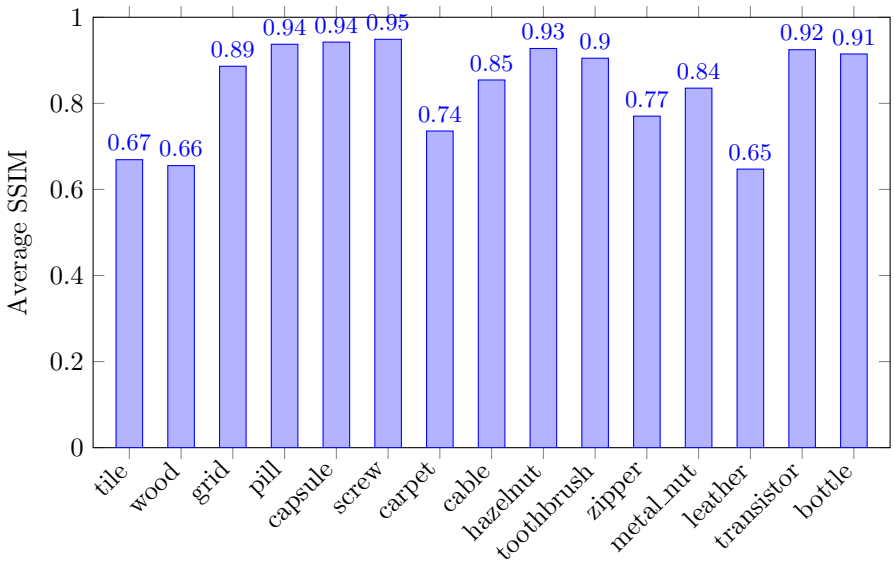
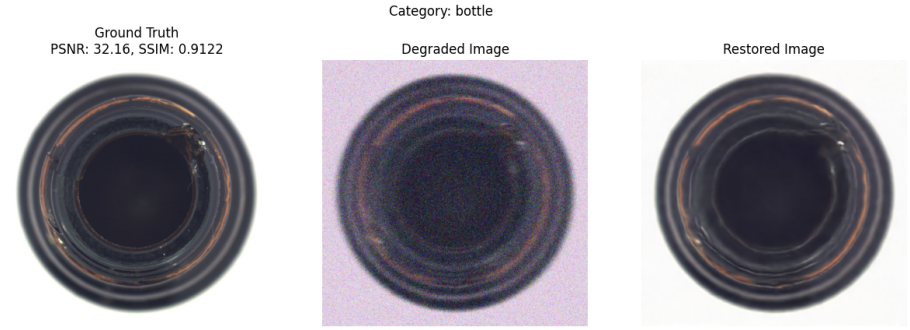
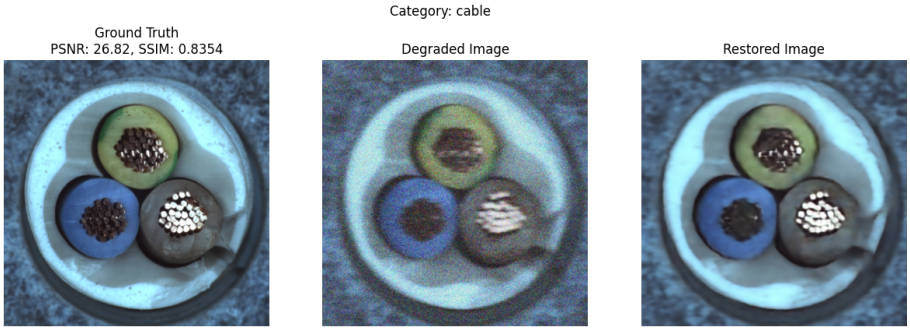
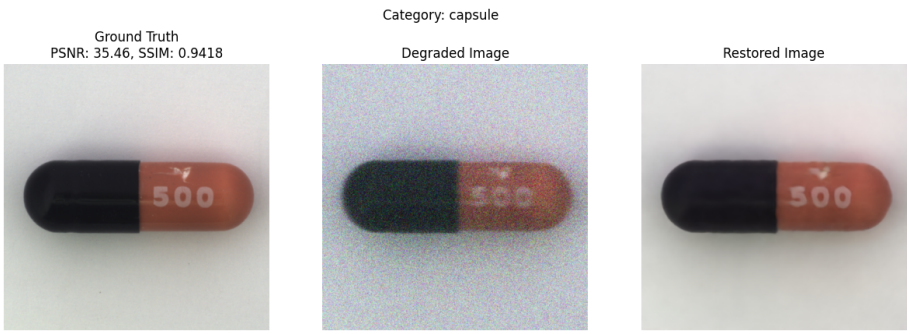


Figure 5: Average SSIM Values for Each Category

5 Sample Outputs for Each Category


Category: bottle
Ground Truth
PSNR: 32.16, SSIM: 0.9122
Degraded Image
Restored Image

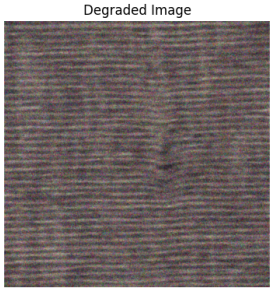

Category: cable
Ground Truth
PSNR: 26.82, SSIM: 0.8354
Degraded Image
Restored Image


Category: capsule
Ground Truth
PSNR: 35.46, SSIM: 0.9418
Degraded Image
Restored Image
500
500
500

Category: carpet

Ground Truth
PSNR: 22.14, SSIM: 0.6814

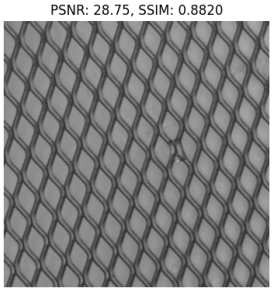
Degraded Image



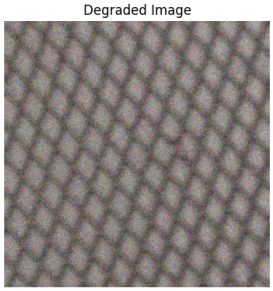
Restored Image

Category: grid

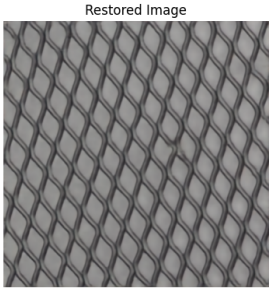
Ground Truth
PSNR: 28.75, SSIM: 0.8820



Degraded Image

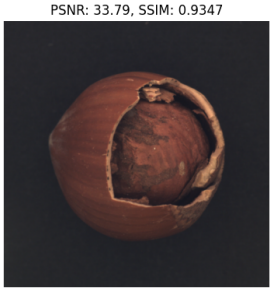


Restored Image



Category: hazelnut

Ground Truth
PSNR: 33.79, SSIM: 0.9347

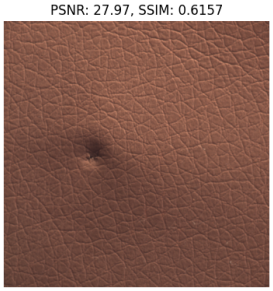


Degraded Image

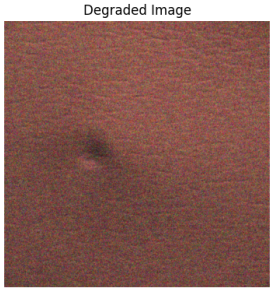
Restored Image

Category: leather

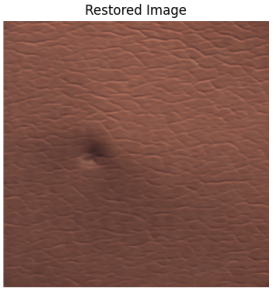
Ground Truth
PSNR: 27.97, SSIM: 0.6157



Degraded Image

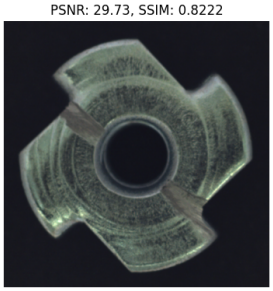


Restored Image

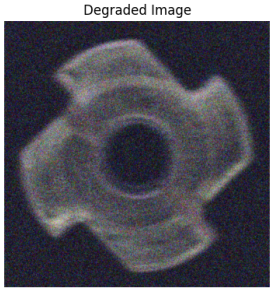


Category: metal_nut

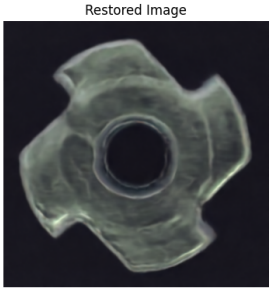
Ground Truth
PSNR: 29.73, SSIM: 0.8222



Degraded Image



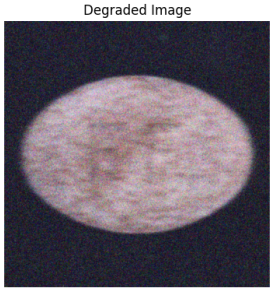
Restored Image



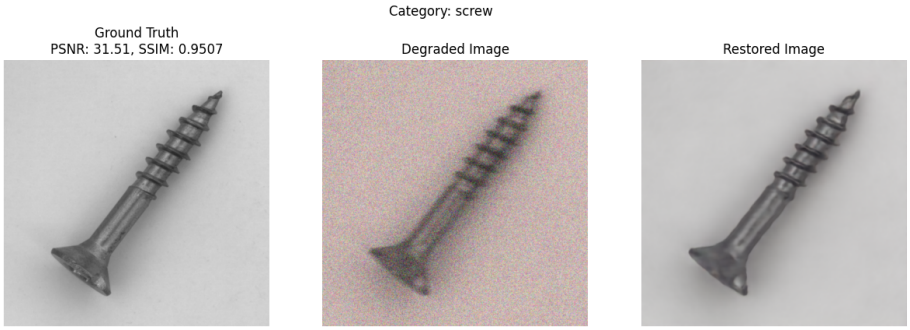
Category: pill

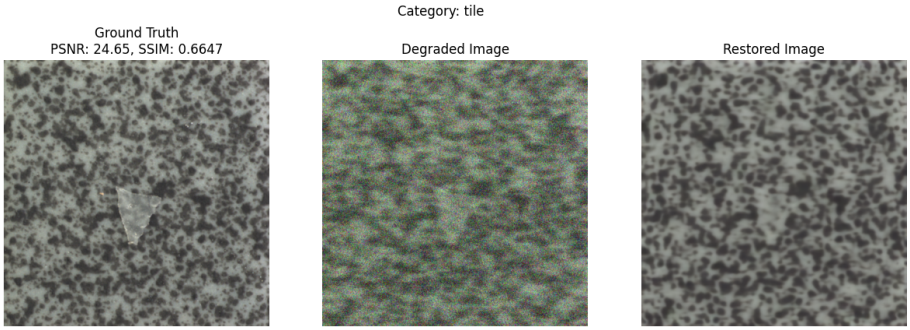
Ground Truth
PSNR: 26.47, SSIM: 0.7788

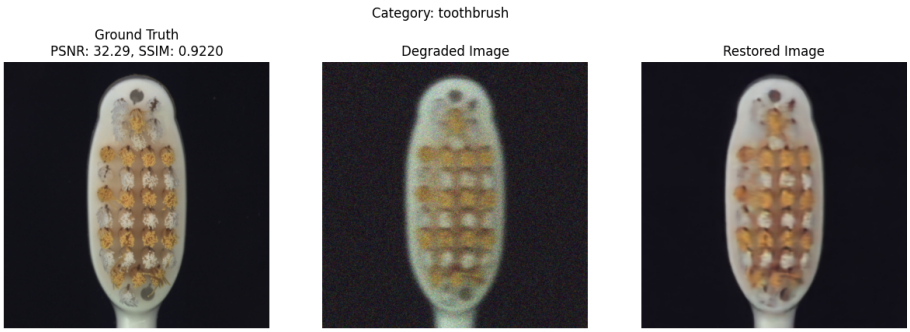
Degraded Image

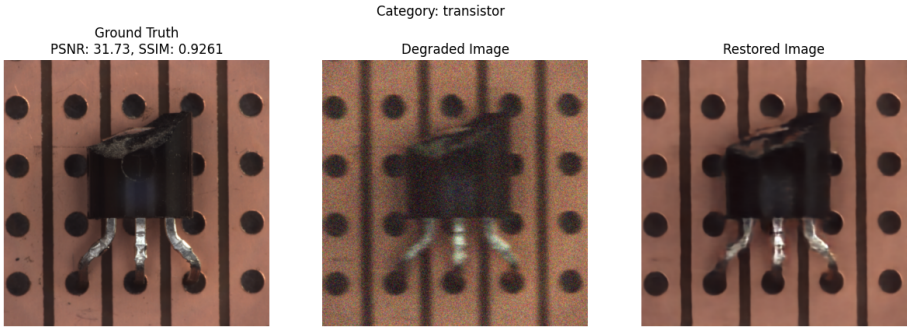


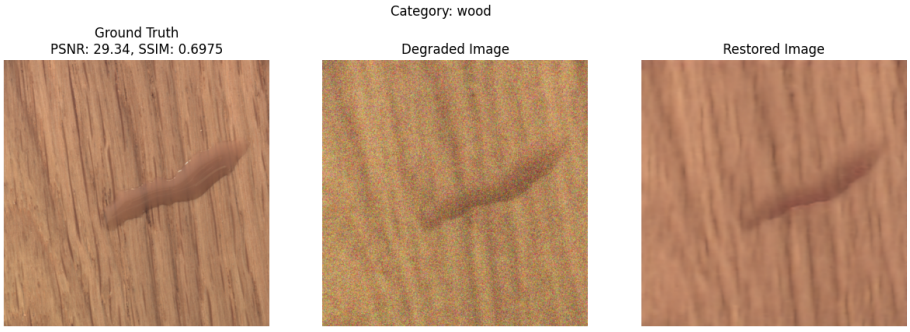
Restored Image

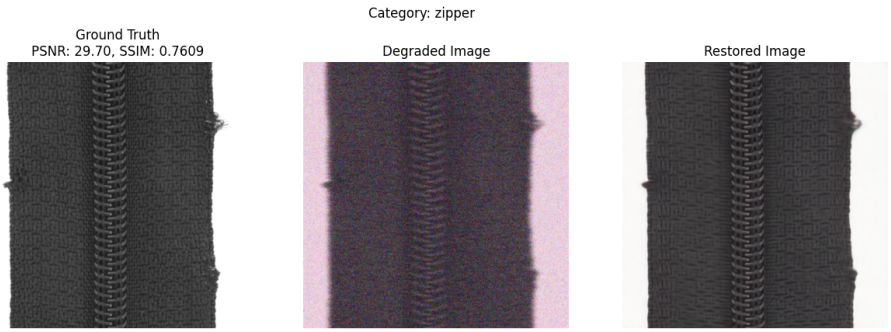

Category: screw
Ground Truth
PSNR: 31.51, SSIM: 0.9507
Degraded Image
Restored Image


Category: tile
Ground Truth
PSNR: 24.65, SSIM: 0.6647
Degraded Image
Restored Image


Category: toothbrush
Ground Truth
PSNR: 32.29, SSIM: 0.9220
Degraded Image
Restored Image


Category: transistor
Ground Truth
PSNR: 31.73, SSIM: 0.9261
Degraded Image
Restored Image


Category: wood
Ground Truth
PSNR: 29.34, SSIM: 0.6975
Degraded Image
Restored Image


Category: zipper
Ground Truth
PSNR: 29.70, SSIM: 0.7609
Degraded Image
Restored Image

6 Github Repository

The project code, model weights, and required packages are available on the Github page:
`https://github.com/Srivatsan6923/EE5179-Deep-Learning-for-Imaging_KLA_Project`

Instructions to run the model, package requirements, and test instructions are also provided in the repository's README.

References

[1] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, Springer, 2015, pp. 234–241.

[2] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 3–19.

[3] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7794–7803.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.