Image Denoising using Non-Local Means and Block Matching and 3D Filtering

Nikhil Viswanath Sivakumar
Department of ECE
Georgia Institute of Technology
Atlanta, United States
nsivakumar33@gatech.edu

Tavish Vats
Department of ECE
Georgia Institute of Technology
Atlanta, United States
tvasts3@gatech.edu

***Abstract*— Image enhancement and its key subsection, image denoising, form a huge part of computer vision and image pre-processing tasks. With a need to growing need of training data for deep learning models and use cases of image visualization being aplenty, this term project for the Digital Image Processing course acts as a platform for students to research on image enhancement. For the given CURE datasets along with SSID and Set-12, this study provides two methods – *Non-Local Means Denoising* and *Total Variation* and an analysis on their performance of various types of distortions. With a comparative evaluation through accuracy metrics and Image Quality Enhancement methods, this detailed report proves the capabilities of the presented enhancement methods for robust and comprehensively distorted images.**

***Keywords*— denoising, quality enhancement, sensor images, smartphone capture, distortion removal**

I. Dataset Review

The smallest among the given datasets is the Set-12 dataset [1]. As the name implies, the dataset contains twelve of the very popular grayscale images used for image processing techniques over the past couple of decades. These include the famous Lena and Cameraman photographs. The size of seven images is of size 256x256, and the rest five are of size 512x512, allowing for swift and simple data loading, augmentation, and processing.

A more modern source of images was provided through the Smartphone Image Denoising Dataset (SIDD) [2], originally containing 30,000 colored images from 10 scenes. Captured under different lighting conditions using five smartphone cameras, these images come as pairs of ground truth and noisy images, containing distortions mainly as the result of varying illumination. The authors chose the SIDD-Small variety dataset having 160 image pairs, acting as an excellent benchmark for denoising techniques.

The Challenging Unreal and Real Environments (CURE) corpus by the OLIVES team at Georgia Tech acts as the major source of images to denoise and enhance. Starting with the Object Recognition dataset (CURE-OR) [3] there are 1 million images of 100 objects spanning across varying sizes, colors, and textures. Mainly captured using smartphones, CURE-OR is a vast dataset of personal, office, and household objects as well as sports, health, and entertainment-based object classes. The dataset was created for 18 challenges to benchmark recognition performance of everyday devices and applications. Examples of the distortions in these challenges include under/over exposure, Salt & Pepper noise, Gaussian blur, and contrast-related effects on RGB-scale and grayscale.

The Traffic Sign Detection dataset (CURE-TSD) [4] and its child dataset, resultant of cropping video frames for traffic sign recognition (CURE-TSR) [4] are the largest datasets in this project. These datasets mirror real-life camera sensing in public environments. scenarios: challenges fittingly reflect these scenarios; Gaussian blurs, darkening, dirty lenses, and distortions from weather and illumination to name a few effects on the image quality. The video dataset totals 5733 video sequences of 1.72 million frames. The recognition dataset is a comprehensive collection of over 2 million images. In summary, the various challenge categories of the CURE datasets test the general robustness of the chosen enhancement algorithms.

Another real-world noisy dataset was chosen as a replacement for the CURE-TSD dataset. This UPoly [5] dataset was proposed to promote studies on image denoising problems using computer vision applications. Hence, this dataset is applied to test two different mathematically intensive non-machine learning filtering methods. Containing 40 different scenes which are cropped to 100 regions of 512x512. Being a purely real-world dataset with natural distortions, this dataset acts as a key indicator of the performance of denoising approaches chosen.

II. Literature Review

Image denoising is an essential task in image processing as it addresses the challenges of removing noise from images without losing important details like edges and textures. It is a complex and ongoing area of research due to its nature as an inverse problem, where solutions are not unique. Conventional denoising methods in the spatial domain, including linear filters like Mean and Wiener filtering, often result in excessive smoothing or blurring of edges. On the other hand, nonlinear filters such as median and bilateral filtering provide improvements but require significant computational resources [6].

An advancement in non-machine learning denoising techniques is the Non-Local-Means (NLM) method. NLM is dependent on using a family of weighted filters based on the non-local self-similarity (NSS) prior [7]. The approach gains its effectiveness and robustness against noise by computing each of the pixel values as weighted averages of similar regions. It is adaptable and effective in preserving image detail, making it an ideal candidate for non-machine learning denoising techniques, especially when dealing with high levels of noise in images.

Looking at transfer domain methods of denoising, Block Matching, and 3D Filtering stands out as a highly effective method. BM3D is an extension of the NLM method. This denoising approach involves stacking similar patches of 3D groups and applying a Weiner Filter. One caveat of this denoising approach is that it tends to add artifacts to images with high levels of noise, especially in the flat areas [7].

In contrast to the non-machine learning denoising techniques, machine learning model-based approaches have had rapid development in the field of image denoising, especially CNN-based model denoising methods. The CNN-based denoising techniques involve learning mappings from degraded and clean image pairs. These models have shown impressive results compared to early adoptions of model-based denoising techniques [7].

Even though machine learning-based denoising techniques would have been the best approach, the time constraints of the project significantly influenced our decision-making. We decided to employ NLM and BM3Ddenoising techniques on the datasets. These traditional non-machine learning denoising techniques do not require the extensive training needed for a machine learning-based approach, particularly for CNN-based denoising. The non-machine learning approaches help us fast-track the process and skip the compiling and training time needed for the large datasets to be evaluated. Moreover, as explained before, NLM and BM3D are known for their effectiveness in a variety of noisy conditions, while being computationally efficient, giving us more time to work on in-depth analysis of these denoising approaches on large datasets.

III. Theory

A. *Non-Local Means Denoising*

As discussed in the literature review section, conventional image smoothing techniques such as Gaussian Blurring acted as baseline methods for removal of noise. Yet, these worked only on small amounts of noise, such as for Salt & Pepper noise. One key feature in the above-mentioned functions was the use of operations (mean, median, weighted average) local to a small neighborhood of pixels. It is important to understand the impact of such simple operations on the nature of noise. A general property of commonly found noises is the zero mean when represented as a random variable. To simply explain this, consider a still camera capturing multiple photos of a certain location when static for a given time. This results in many frames of the same scene. If these images are averaged out using any coding implementation, the resultant frame will have a reduction in noise compared to the first few frames captured. Unfortunately, this simple experiment does not extend to scene motions and stability of camera captures. Very often, such an experiment would result in only one or a few noisy images.

Hence, to improve on this idea, a starting point is to obtain a larger set of similar images to average out the noise. If we take a small 5x5 window in one of the frames, the chances of finding a particular patch outside such a small window is high. The same patch might be in a small neighborhood around the window. Thus, taking similar patches together and averaging would reduce the noise distortion. Essentially, taking a pixel and setting a window around it and searching for similar windows in the image. Then we average all these windows and replace the chosen pixel with the averaging result. This method is termed *Non-Local Means (NLM)* denoising. Evidently it is slightly more complex and time consuming to iterate through the frame. However, the resultant output will be denoised in a much better manner.

The NLM algorithm can be defined as a simple function. Suppose *X is an* image with discrete pixels, and *p* and *q* are two points within the image. The NLM function [ ] is given by:

$$u(p) = \frac{1}{C(p)} \sum_{q \in X} v(q) f(p,q)$$

where *v(q)* is the unfiltered value of the image at point *q*. *C(p)* is given by *C(p)* = $\mathrm{C}(\mathrm{p}) = \sum_{q \in X} f(p,q)$.

The weighing function can be Gaussian, i.e., a normal distribution with a mean *B(p)* and variable standard deviation *h*.

$$f(p,q) = e^{-\frac{|B(q)^2 - B(p)^2|}{h^2}}$$

where *B(p)* is given by:

$$f(x) = \frac{1}{|R(p)|} \sum_{i \in R(p)} v(i)$$

Where *R(p)* X is a square region of pixels surrounding *p* and |*R(p)*| is the number of pixels in region *R*.

B. *BM3D*

Block-matching is a particular matching approach used for motion estimation and compression in videos. Block-matching finds block of images similar to a given image, which is stacked into a 3D array, which is called a" group". This process is called grouping and is the first step in a broader algorithm termed Block-matching and 3D filtering (BM3D), shown in Fig. 1 [8].

To deal with these three-dimensional groups, filtering is performed through the main step in this algorithm – collaborative filtering. The basic idea of it is to perform filtering on every fragment group by applying a linear transform which is a dimension higher than the fragment in each group. Later, a transform-domain shrinkage is applied, followed by an inverse linear transform to reproduce all the filtered fragments. Since the individual blocks are 2D image fragments, we require a 3D linear transform on this 3D group.

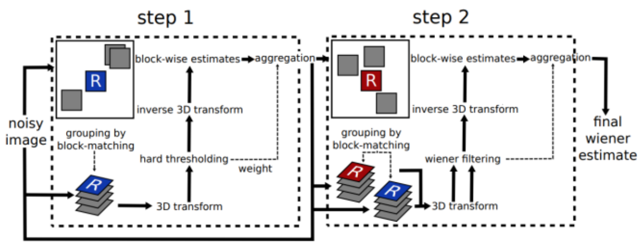


Figure 1: Scheme of the BM3D algorithm.

While applying collaborative filtering, the individual blocks will appear overlapping and as a result, different estimates computed need to be combined. This is performed through aggregation technique, which is an averaging procedure to handle this redundancy.

The core of the BM3D algorithm is the shrinkage step on the spectrum of grouped noisy blocks. A generic shrinkage for a given transform-domain coefficient of the group can be expressed as [8]:

$$s_{i,j}^{x_1,\dots,x_M} \longmapsto \alpha_{i,j} s_{i,j}^{x_1,\dots,x_M}$$

where $\alpha_{i,j}$ is a shrinkage attenuation factor depending on the noise statistics $s_{i,j}^{x1,\dots xM}$.

The common BM3D algorithm uses hard-thresholding in the first denoising stage followed by an empirical Wiener filter in the second denoising stage. In hard-thresholding, the shrinkage is performed by setting spectrum coefficients lower than a threshold to zero, since most of the coefficients are composed of noise. In Wiener filtering, the attenuation coefficients of the transfer function are computed from the previous estimate and from the variance of noise spectrum coefficients. The variance is given by:

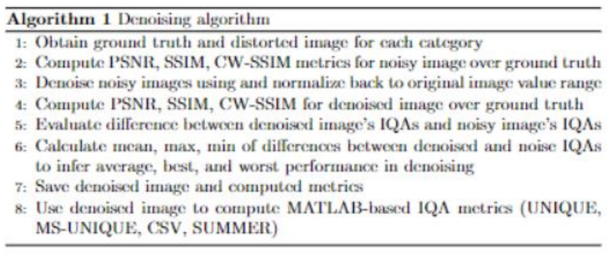
$$\alpha_{i,j}^{\text{wie}} = \frac{\| \langle [\hat{y}_{x_1}^{\text{HT}}; \cdots ; \hat{y}_{x_M}^{\text{HT}}] , b_i^{dD} \otimes b_j^{\text{NL}} \rangle \|^2}{\| \langle [\hat{y}_{x_1}^{\text{HT}}; \cdots ; \hat{y}_{x_M}^{\text{HT}}] , b_i^{dD} \otimes b_j^{\text{NL}} \rangle \|^2 + \mu^2 v_{i,j}^{x_1,\dots,x_M}}$$

where y^{HT} is the estimate of y obtained from hard-thresholding stage, b_i^{dD} is the i-th basis function of the d-dimensional spectrum, and v corresponds to variance of spectrum coefficients. μ^2 is a scaling factor included due to aggregation to influence the bias-variance ratio we wish to minimize through the Wiener filter. After this step, the aggregation part combines obtained estimates into a buffer using aggregation weights, where a larger weight is given to blocks with less residual noise, to improve the denoising. Finally, the residual noise variance for blocks of the group is calculated with new, block-specific aggregation weights. In other words, we compute the sum of residual variance within each block after shrinkage by applying the inverse transform, and then we invert the value to obtain the final aggregation weights. The general idea behind weighted aggregation is to improve quality of the final estimate and reduce the visibility of artifacts.

Collaborative filtering can reveal fine details shared by grouped blocks while preserving unique features of each individual block. It acts as an excellent denoising technique, which is computationally less intensive than existing clustering techniques.

IV. Implementation Approach

Based on the theoretical understanding described in the above section, the Non-Local Means and the Block-matching and 3D filtering techniques are implemented using Python. A basic algorithm of how the denoising techniques were performed is described below.



```
Algorithm 1 Denoising algorithm
1: Obtain ground truth and distorted image for each category
2: Compute PSNR, SSIM, CW-SSIM metrics for noisy image over ground truth
3: Denoise noisy images using and normalize back to original image value range
4: Compute PSNR, SSIM, CW-SSIM for denoised image over ground truth
5: Evaluate difference between denoised image's IQAs and noisy image's IQAs
6: Calculate mean, max, min of differences between denoised and noise IQAs
   to infer average, best, and worst performance in denoising
7: Save denoised image and computed metrics
8: Use denoised image to compute MATLAB-based IQA metrics (UNIQUE,
   MS-UNIQUE, CSV, SUMMER)
```

Non-Local Means denoising technique was implemented in Python using Scikit-Image's [9] restoration module *denoise_nlmeans*. A custom python function which first uses the Scikit-Image's *estimate_sigma* function, then uses a defined patch of size 5 and distance 6. The patch distance corresponds to the maximal distance in pixels to search patches used for denoising. The *estimate_sigma* function provides a good starting point for setting the *h* (variable standard deviation for Gaussian weighing function) and *sigma* parameters. *h* controls the decay in patch weights as a function of distance between patches. *h* was set as the mean of the output of *estimate_sigma* scaled by 1.15, and the algorithm was run in the fast mode, which uses a faster algorithm employing uniform spatial weighting on the patches.

BM3D was implemented through Python by installing the available *bm3d* [10] wrapper for stationary correlated noise. Then, by calling the *bm3d* function of the library, images were denoised. The input parameters were the noisy image, a *sigma_psd* value, and a *stage_arg* argument. The *sigma_psd* corresponds to Noise Power Spectral Density and was set as 0.13 after experimenting on a range of values. *stage_arg* was set as all stages, which means both hard-thresholding and Wiener filtering.

Evaluation of performance of denoising metrics were based on seven Image Quality Assessment metrics, which are explained in the Analysis section of this report. The first three metrics – PSNR, SSIM, CW-SSIM – were implemented in Python. PSNR and SSIM were obtained from Scikit-Image, an image processing library for Python. For CW-SSIM, a module termed *pyssim* was implemented in Python [11]. The rest four metrics were obtained from OLIVES GitHub repositories and were implemented on MATLAB v2023b.

All datasets were downloaded from official sources. Contents of Set-12 dataset was the only gray-scale image dataset used. For SIDD, the small version was chosen from their official website. UPoly dataset was obtained from the authors' official GitHub repository. CURE-OR and CURE-TSR were provided by the course instructor from OLIVES Georgia Tech GitHub repositories.

For Set-12 images, two types of noises were added – Gaussian noise and Salt & Pepper noise, both with 0.15 standard deviation and *p-value* 0.001.

CURE-OR contained 9 different distortion challenges, each for colored and grayscale images, of 5 different challenge levels. These were captured using 5 cameras and objects were placed in 5 different backgrounds. The implementation was done for only colored images, in one camera for all different challenges and challenge levels. For the sake of this report, only limited distortion challenges are mentioned with the smallest, the medium, and highest distortion levels mentioned purely to summarize the performance of two denoising techniques.

Later, the generated denoised images for CURE-OR dataset for both NLM and BM3D methods were evaluated for object recognition on a ResNet50 model from TensorFlow Keras Applications [21] using Google Colab on Tesla4 GPU. Pre-trained ImageNet weights were used. The top-3 recognized object predictions were reported.

V. Results and Analysis

A. *Set-12*

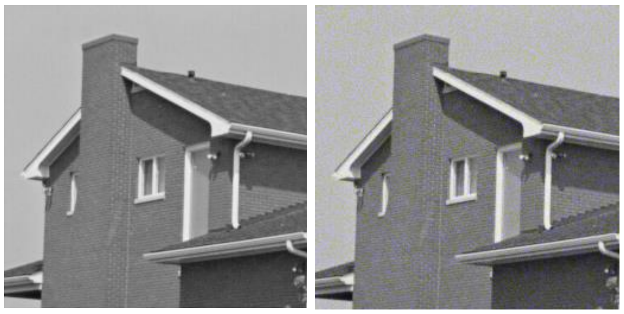


Figure 2: NLM and BM3D denoised images for Gaussian Noise

Table Head	**NLM for Gaussian Noise**		
	PSNR	***SSIM***	***CW-SSIM***
Noisy	21.26	0.76	0.38
Denoised	30.96	0.86	0.93

Table 1: Python IQA for NLM on Set-12

Table Head	**NLM vs BM3D for Gaussian Noise**			
	UNIQUE	***MS-UNIQUE***	***CSV***	***SUMMER***
NLM	0.88	0.92	0.98	4.35
BM3D	0.85	0.90	0.98	4.17

Table 2: MATLAB IQA for NLM & BM3D on Set-12

Figure 2 shows the denoised output images for NLM and BM3D techniques for one image with Gaussian noise added. The quality of the image seems to be intact, with no noisy grains left. Looking from the naked eye, both outputs seem similar, leading to performance analysis through IQAs. The tables below the image show NLM improving PSNR by 9 dB, SSIM by 10% and CW-SSIM by 55%. Comparing the two denoising techniques, NLM edges out BM3D when observing metrics such as UNIQUE and SUMMER.

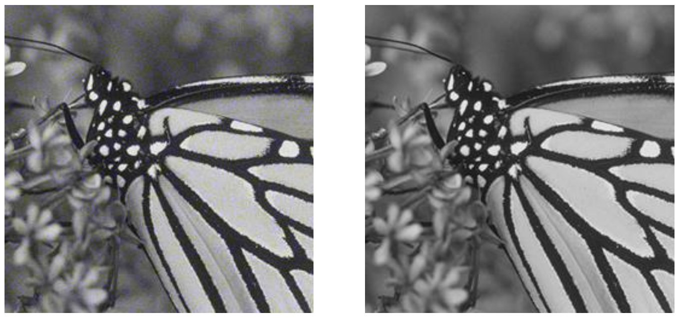


Figure 3: NLM and BM3D denoised images for S&P noise

Table Head	**BM3D for Gaussian Noise**		
	PSNR	***SSIM***	***CW-SSIM***
Noisy	21.25	0.80	0.44
Denoised	28.20	0.89	0.98

Table 3: Python IQA for BM3D on Set-12

Table Head	**NLM vs BM3D for Gaussian Noise**			
	UNIQUE	***MS-UNIQUE***	***CSV***	***SUMMER***
NLM	0.90	0.92	0.98	4.30
BM3D	0.88	0.90	0.98	4.15

Table 4: MATLAB IQA for NLM & BM3D on Set-12

Next, we test out denoising on Salt & Pepper noise. For the second image, PSNR improves by 7dB, with SSIM increase by 8% and CW-SSIM by 54% for BM3D. Though a good improvement, NLM still performs slightly better in terms of MS-UNIQUE and SUMMER metrics.

Overall, for all 12 ground truth images, with both Gaussian noise and Salt & Pepper noise, both denoising techniques perform very well and enhance quality of image. Overall performance shows NLM is slightly better, though both techniques give high enhancement. This could be due to images being grayscale and less intensive in intricacies. Since this dataset consisted of smaller pixel dimensions (256 or 512), denoising was quicker and less error-prone.

B. SIDD

The SIDD (Smartphone Image Denoising Dataset) contains various instances of images taken from Google, iPhone, Samsung, Motorola, and LG phone cameras. For our analysis, we chose to use the SIDD small dataset containing 160 image pairs of noisy and ground truth images. This dataset represents the noise from the smartphone cameras under various conditions (different ISO levels, shutter speeds, and lighting conditions). Following a similar approach to how we evaluated performance of denoising techniques in CURE-TSR, we find the mean, maximum, and minimum difference of IQA metrics to show the average, worst-, and best-case performance respectively for each denoising method on the SIDD dataset. The IQA metric evaluation is shown in Table 5.

BM3D	PSNR (dB)	SSIM	CW-SSIM	UNI QUE	MS-UN IQUE	CSV	SUM MER
mean	5.79	0.27	0.04	0.17	0.1	0	0.03
min	-0.2	-0.03	-0.03	-0.15	-0.08	0	-1.25
max	9.7	0.57	0.17	0.39	0.44	0.01	0.21
NLM	PSNR (dB)	SSIM	CW-SSIM	UNI QUE	MS-UN IQUE	CSV	SUM MER
mean	6.36	0.26	0.	0.18	0.11	0	0.05
min	-1.21	-0.07	-0.03	-0.12	-0.06	0	-0.3
max	9.76	0.53	0.18	0.48	0.51	0.01	0.18

Table 5: IQA Metric Performance for SIDD Dataset

Looking at the SIDD dataset, we can get an overall idea of how BM3D and NLM are performing on these noisy smartphone images. From the table, we can infer that all the metrics show a positive performance and big improvements in especially the PSNR, SSIM, UNIQUE, and MS-UNIQUE IQA metrics for both the denoising techniques. It seems that both BM3D and NLM perform well when dealing with sensor noise or noise due to low lighting conditions. BM3D performs well, especially for images with higher ISO levels. When doing a comparative analysis of the two denoising techniques, we see that both the PSNR and SSIM perform well and at a similar level. For CW-SSIM, we see that NLM may perform better in some cases as it has a better average and max difference compared to BM3D. This is because of NLM ability to preserve the local patterns and textures, which is important in the context of calculating CW-SSIM. Same goes for metrics UNIQUE and MS-UNIQUE as they are better for NLM compared to BM3D. This is because of NLM’s emphasis on maintaining local patch similarities, which is important in the context of UNIQUE and MS-UNIQUE as they compute image quality detail and textures. Overall, the results for the SIDD dataset when using NLM and BM3D dataset are promising and show good results as evident from Table 1.

C. CURE-OR

We performed the most extensive denoising for this dataset and CURE-TSR, with over 10 challenge distortions and 5 challenge levels. The distortions in CURE-OR drastically impacted the quality of the object, such that the PSNR of the noisy image with the ground truth image was less than 10 dB for most cases. Hence, it was imperative to enhance these images containing household and office objects.

Table 6 shows a comparison for NLM and BM3D denoising techniques salt & pepper noise distorted of level 3. Consistently across different objects, BM3D performs better and enhances the image significantly. Though NLM denoising works, it is only slightly better than the initial noisy image.

BM3D	PSNR (dB)	SSIM	CW-SSIM	UNI QUE	MS-UN IQUE	CSV	SUM MER
001	10.12	0.20	0.43	0.958	0.940	0.12	3.692
005	10.01	0.21	0.44	0.959	0.939	0.04	3.699
007	9.74	0.19	0.46	0.958	0.939	0.09	3.692
NLM	PSNR (dB)	SSIM	CW-SSIM	UNI QUE	MS-UN IQUE	CSV	SUM MER
001	15.74	0.36	0.53	0.051	0.132	0.84	1.541
005	15.68	0.37	0.53	0.049	0.133	0.86	1.546
007	15.75	0.37	0.54	0.052	0.131	0.89	1.548

Table 6: IQA Metric Performance for CURE-OR Dataset

A clear distinction between NLM denoised and BM3D denoised images are visible. This trend was visible for multiple distortion challenges. Through this we identify that block matching for high detailed large dimensional three channel images. This was proved later in the object recognition performance evaluation as well.

D. CURE-TSR

The CURE-TSR (Challenging Unreal and Real Environments for Traffic Sign Recognition) contains and diverse set of unreal and real data. The real and unreal datasets are format and organized based on the traffic sign type, the challenge type, and the level of challenge type. The challenge types for CURE-TSR include: Decolorization (01), Lens blur (02), Codec error (03), Darkening (04), Dirty lens (05), Exposure (06), Gaussian blur (07), Noise (08), Rain (09), Shadow 10), Snow (11), and Haze (12). Each of these challenge types has a level from 1 (low) to 5 (severe).

We conducted analysis on the dataset by calculating the IQA metrics described in the project for noisy image and the denoised image using BM3D and NLM techniques. To understand how good the denoised images quality is for a particular challenge type, we calculate the mean, minimum, and maximum difference in the IQA metrics for each challenge type, considering each the challenge level and all the 14 traffic sign types. The difference in the IQA metrics is calculated by subtracting IQA metrics of noisy images from the IQA metrics of denoised images obtained using the denoising techniques. Table 6 and Table 7 show the results obtained for mean, minimum, and maximum difference in IQA metrics for the real and unreal data respectively with the rows representing a few challenges type and the columns representing the 7 IQA metrics described in the project guidelines. Each cell contains three numbers in the order: mean, minimum, and maximum difference.

BM3D	**PSNR (dB)**	**SSIM**	**CW-SSIM**	**UNIQUE**	**MS-UNIQUE**	**CSV**	**SUMMER**
01	0.01, -0.1, 0.15	0, 0, 0.01	0, -0.03, 0.01	0, -0.15, 0.14	0, 0.09, 0.02	0, 0, 0	-0.05, -1.4, 0.16
06	0, -0.01, 0.05	0, -0.01, 0.01	-0.01, -0.1, 0.02	-0.03, -0.45, 0.05	-0.02, -0.35, 0.07	0, -0.01, 0	0.06, -0.3, 1.97
08	0.4, 0, 3.05	0.01, -0.01, 0.03	0, -0.07, 0.03	0, -0.3, 0.16	0, -0.2, 0.07	0, 0, 0.01	-0.03, -0.92, 0.42
NLM	**PSNR (dB)**	**SSIM**	**CW-SSIM**	**UNIQUE**	**MS-UNIQUE**	**CSV**	**SUMMER**
01	0.01, -0.39, 0.37	0, -0.01, 0	0, 0, 0	0, -0.05, 0.01	0, -0.01, 0.01	0, 0, 0	0.05, -0.21, 0.28
06	0, -0.01, 0.01	0, -0.01, 0	0, 0, 0	0, -0.01, 0.01	0, -0.01, 0.01	0, 0, 0	0.01, -0.11, 0.46
08	0.18, -0.01, 1.28	0.01, 0, 0.05	0, 0, 0.03	0, -0.05, 0.04	0, -0.01, 0.04	0, 0, 0	0.01, -0.4, 0.46

Table 6: Real Data IQA Performance on CURE-TSR

BM3D	**PSNR (dB)**	**SSIM**	**CW-SSIM**	**UNIQUE**	**MS-UNIQUE**	**CSV**	**SUMMER**
01	0.02, -0.01, 0.21	0, 0, 0	0, -0.01, 0	0, -0.01, 0.01	0, 0, 0	0, 0, 0	0, -0.01, 0.02
06	0, -0.01, 0.03	0, 0, 0.02	0, -0.03, 0.04	0, -0.14, 0.08	0, -0.11, 0.05	0, -0.01, 0	-0.03, -1.79, 0.55
08	0.43, 0, 3.24	0.01, 0, 0.08	0, -0.02, 0.04	0, -0.02, 0.08	0, -0.01, 0.02	0, 0, 0.01	-0.03, -0.65, 0.3
NLM	**PSNR**	**SSIM**	**CW-SSIM**	**UNIQUE**	**MS-UNIQUE**	**CSV**	**SUMMER**
01	0, -0.18, 0.06	0, -0.01, 0	0, 0, 0.01	0, -0.01, 0.01	0, 0, 0	0, 0, 0	0, -0.02, 0.11
06	0, -0.04, 0.01	0, -0.01, 0	0, -0.01, 0	0, 0, 0	0, -0.04, 0	0, 0, 0	0.01, -0.4, 0.44
08	0.22, -0.01, 3.39	0.01, 0, 0.1	0, -0.01, 0.03	0, -0.02, 0.05	0, 0, 0.02	0, 0, 0.01	0, -0.32, 0.73

Table 7: Unreal Data IQA Performance on CURE-TSR

Looking at the CURE-TSR evaluation Tables 6 and 7, we see a common theme in terms of the average, minimum, and maximum differences of all the IQA metrics. We can see that for both the real and unreal data the IQA metrics performance has minimal to no improvement in the image quality as evident from both the Table values. In the Table, we chose to show only a subset of the results, which are for challenge type Decolorization, Exposure, and Noise. The reason behind this is to showcase the limitations of our denoising techniques on certain challenges. The complex distortion of challenge types such as exposure, decolorization, and even other challenge types such as rain, snow, haze, etc. don't align well with the types of noise that traditional non-machine learning denoising techniques like BM3D and NLM are optimized to handle. Hence, the results obtained for exposure and decolorization are justified. On the other hand, we notice that for challenge type of noise, we see that for both real and unreal data, the PSNR and SSIM show an average increase. This result shows that for certain challenge types in CURE-TSR where it is dealing with basic noise and/or blur, the denoising technique is improving the image quality to some extent. Overall, as evident from the table, the two denoising techniques show almost no improvement in image quality and some of the IQA metrics have a negative value, which means that the denoising techniques is not working as expected on the complex set of challenges in the CURE-TSR dataset.

E. UPoly

The UPoly (Real World Noisy Images Dataset) is a unique dataset that contains high quality images taken from some of the best camera brands in the world like Canon, Nikon, and Sony. The dataset is relatively small: only 40 scenes. Due to the really high pixel resolution and dimensions of the original image, the dataset provides 100 cropped versions of the original images in RGB format of shape (512x512) in pairs of ground truth and noisy images. For our analysis, we follow the same approach as previous dataset of calculating the average, minimum, and maximum difference in IQA metrics to show how well our two denoising techniques perform on this dataset. The IQA metrics evaluation is shown in Table 2.

BM3D	PSNR (dB)	SSIM	CW-SSIM	UNIQUE	MS-UNIQUE	CSV	SUMMER
mean	2.25	0.07	0.03	0.16	0.07	0	0.08
min	-0.3	-0.03	-0.02	-0.06	-0.04	0	-0.09
max	5.85	0.27	0.1	0.4	0.14	0.01	0.24
NLM	PSNR (dB)	SSIM	CW-SSIM	UNIQUE	MS-UNIQUE	CSV	SUMMER
mean	1.9	0.07	0.04	0.16	0.08	0	0.06
min	-0.47	-0.04	-0.02	-0.06	-0.04	0	-0.11
max	4.92	0.249	0.12	0.45	0.15	0.01	0.18

Table 8: IQA Metric Performance for UPoly Dataset

F. Performance Accuracy for Object Recognition

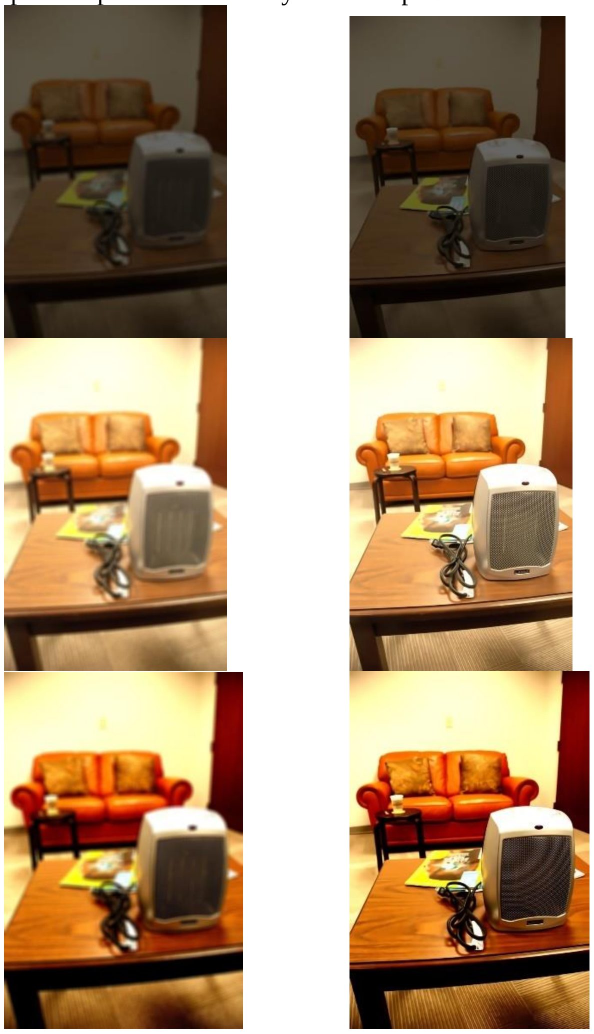
Recognition accuracy on the denoised images were run for a pre-trained ResNet model. The images below the denoising outputs for 3 types of distortions over a CURE-OR enhancement output. We identified the prediction accuracies of recognizing objects in the denoised image, with comparison to ground truth and noisy image across few distortions.

For an example image of a space heater placed on a table, we find the trends in prediction for denoised images. Given a ground truth prediction of 82%, the underexposure distorted image scored 94%, the overexposure scored 30%, salt & pepper noise scored just 20% and contrast scored a high 81%.To show a decent amount of clarity and conciseness in the report, these 4 challenges were chosen as they differ in property of distortion, and to view images properly we show the Level-1 challenge.

NLM denoised images score poorly for underexposure (30%) and overexposure (18%). No top-3 finish for contrast image, though it is the denoised output. Enhancing salt & pepper enhancing was successful (76%).

BM3D performs significantly better, with underexposure and overexposure scores of 77% 66%, but still poorer than the noisy image itself. The best performance for contrast distortion was for the BM3D denoised image with 76% accuracy, just 5% short of the noisy image score. For salt & pepper noise, the prediction of BM3D (94%) is better than noisy image (82%) and even the ground truth image (92%).

This is reflected in the quality of the denoised images shown. BM3D's output is uniformly much smoother, and sharper compared to the blurry NLM output.



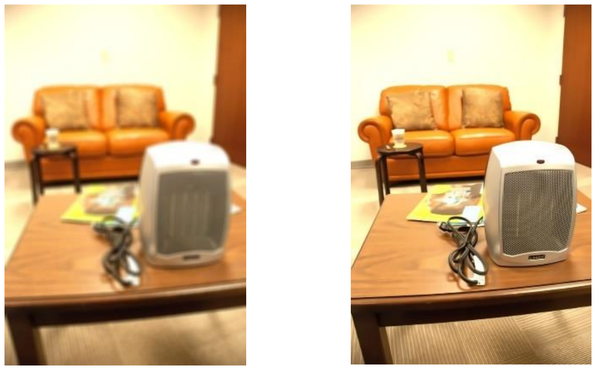


Figure 4: Comparison of denoising for NLM (left) and BM3D (right) for underexposure, overexposure, contrast, and salt & pepper noise distorted challenge in level-1 CURE-OR.

Conclusion & Future Work

In this project, we were able to apply and learn how traditional non-machine learning denoising techniques like BM3D and NLM perform on various challenging datasets including Set12, SIDD, CURE-OR, CURE-TSR, UPoly datasets. Throughout the project, we demonstrated the reliability and performance of these denoising techniques by finding the difference in IQA metrics and calculating the average, worst-, and best-case performance corresponding to the mean, min, and max differences in the 7 IQA metrics. The IQA metrics we used for analysis are PSNR, SSIM, CW-SSIM, UNIQUE, MS-UNIQUE, CSV, and SUMMER, which were all implemented on Python and MATLAB, and they revealed insightful trends and performance of these techniques in diverse scenarios presented in the datasets.

Some key observations from our project were the varying performance of BM3D and NLM across various datasets. BM3D showed great results in reducing Gaussian-like noise, which is evident in the high performance it achieved for the UPoly and SIDD datasets. BM3D's patch-based collaborative filtering method significantly improved and reduced noise for those datasets. On the other hand, NLM also showed its strengths in preserving the local textures and patterns and its emphasis on maintaining local path similarities, which was evident in both those datasets.
However, we faced numerous challenges throughout the project. The primary constraint we faced was time and computational resources. The big dataset sizes and the computational power needed to test other denoising methods required a lot of computational resources. These limitations impacted our choice of denoising methods and our ability to explore more sophisticated denoising methods that would offer better performance for datasets like CURE-TSR and CURE-OR.

[1] K. Zhang, W. Zuo, Y. Chen, D. Meng and L. Zhang, "Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising," in *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142-3155, July 2017, doi: 10.1109/TIP.2017.2662206.

[2] Abdelrahman Abdelhamed, Lin S., Brown M. S. "A High-Quality Denoising Dataset for Smartphone Cameras", IEEE Computer Vision and Pattern Recognition (CVPR), June 2018..

[3] Dogancan Temel, Jinsol Lee, Ghassan AlRegib, October 13, 2019, "CURE-OR: Challenging Unreal and Real Environment for Object Recognition", IEEE Dataport, doi: https://dx.doi.org/10.21227/h4fr-h268.

[4] Dogancan Temel, Gukyeong Kwon, Mohit Prabhushankar, Ghassan AlRegib, October 10, 2019, "CURE-TSR: Challenging Unreal and Real Environments for Traffic Sign Recognition", IEEE Dataport, doi: https://dx.doi.org/10.21227/n4xw-cg56.

[5] Jun Xu, Hui Li, Zhetong Liang, David Zhang, and Lei Zhang. Real-world Noisy Image Denoising: A New Benchmark. https://arxiv.org/abs/1804.02603, 2018.

[6] Fan, L., Zhang, F., Fan, H. et al. Brief review of image denoising techniques. Vis. Comput. Ind. Biomed. Art 2, 7 (2019). https://doi.org/10.1186/s42492-019-0016-7

[7] Fan, L., Zhang, F., Fan, H. et al. Brief review of image denoising techniques. Vis. Comput. Ind. Biomed. Art 2, 7 (2019). https://doi.org/10.1186/s42492-019-0016-7

[8] Y. Mäkinen, L. Azzari and A. Foi, "Collaborative Filtering of Correlated Noise: Exact Transform-Domain Variance for Improved Shrinkage and Patch Matching," in IEEE Transactions on Image Processing, vol. 29, pp. 8339-8354, 2020, doi: 10.1109/TIP.2020.3014721.

[9] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu and the scikit-image contributors. scikit-image: Image processing in Python. PeerJ 2:e453 (2014) https://doi.org/10.7717/peerj.453

An Analysis and Implementation of the BM3D Image Denoising Method Marc Lebrun1. https://doi.org/10.5201/ipol.2012.l-bm3d