# Assignment 2

February 12, 2025

# 1 Pixel-wise contrast enhancement

## 1.1 Gray scale Image

```python
def gamma_transform(image, gamma):
    # Normalize to [0,1]
    image = image.astype(np.float32) / 255.0
    # Apply gamma correction
    corrected = np.power(image, gamma)
    # Convert back to [0,255]
    return (corrected * 255).astype(np.uint8)
```

A gamma=1 is equals to the original gray-scale picture, a gamma=0.5 makes dark regions look brighter, and a gamma=2.0 darkens bright regions. See figure 1.

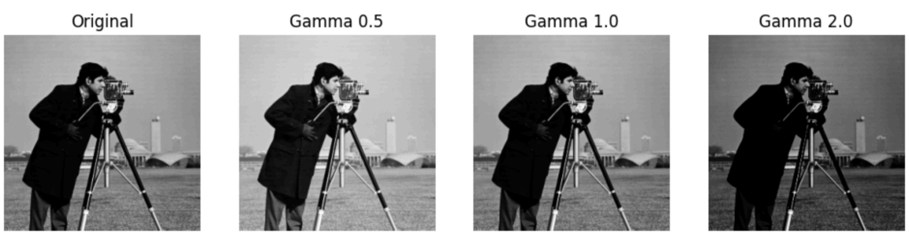



Figure 1: Pixel-wise contrast enhancement on a gray scale picture

## 1.2 Color Image - RGB correction

See figure 2.

```python
def gamma_correct_hsv(image, gamma):
    hsv_image = color.rgb2hsv(image)
    hsv_image[..., 2] =
      gamma_transform(
        (hsv_image[..., 2] * 255).astype(np.uint8), gamma) / 255.0
    return (color.hsv2rgb(hsv_image) * 255).astype(np.uint8)
```

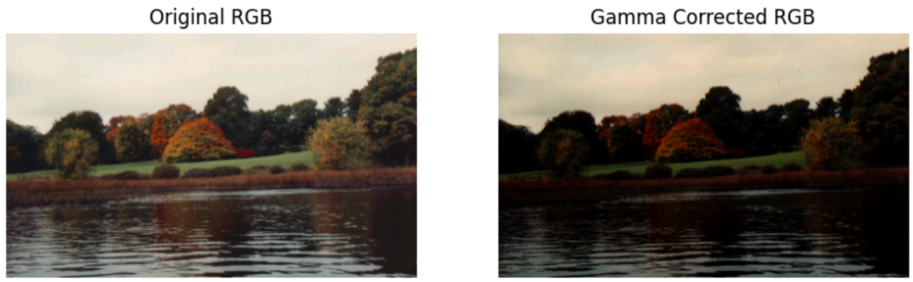



Figure 2: Pixel-wise contrast enhancement on a gray scale picture

### 1.3 Color Image - HSV color representation

```
def gamma_correct_hsv(image, gamma):
    hsv_image = color.rgb2hsv(image)
    hsv_image[..., 2] =
      gamma_transform((hsv_image[..., 2] * 255).astype(np.uint8), gamma) / 255.0
    return (color.hsv2rgb(hsv_image) * 255).astype(np.uint8)
```

See figure 3. The result of the RGB Correction is a little more brighter on the bright part, and thus look less natural. This may because the RGB Correction alters all channels independently, and the colors may be distorted. The HSV Correction modifies only the brightness while preserving colors, so it looks better.


Figure 3: Pixel-wise contrast enhancement on a gray scale picture

## 2 Reverb convolution

### 2.1 Plot laugh2.wav

See figure 5. The orange curve represents right sound track, and the blue curve represents the left sound track. The height of the curve represents the amplitude of this audio channel over time, with x-axis as time and y-axis as amplitude. The sample rate represents how many samples per second are used to represent the sound.

### 2.2 Reverb

Reverb simulates sound reflections in space, making it feel more spacious and echoing. With the convolution of Clap, the reverb mix the clap and the origin laughter, and it sounds like the laughter

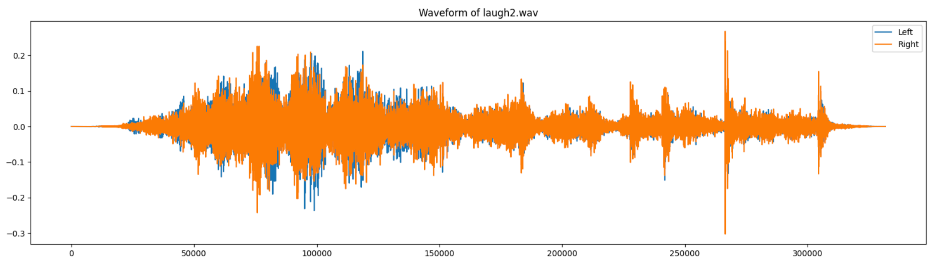
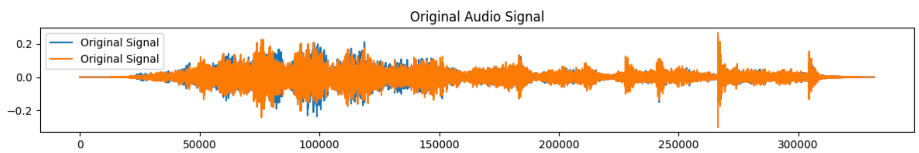
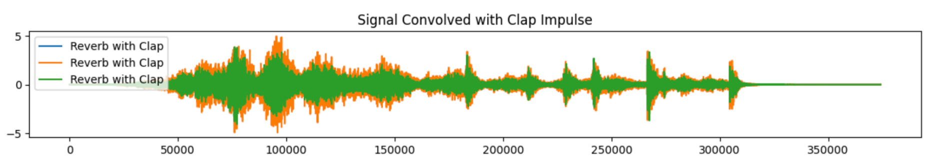



Figure 4: Plot laugh2.wav

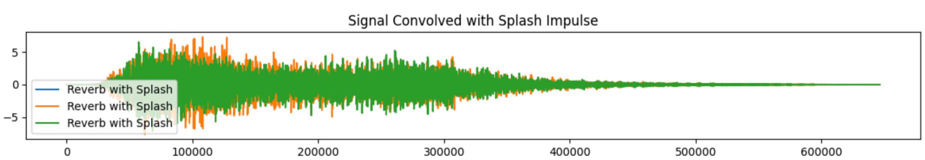
is mixed with claps in a hall. See **??**. With the convolution of Splash, the laughter sounds like coming from the outer space. See 5(c).




(a) Original Signal




(b) Response Signal, Clap




(c) Response Signal, Splash

Figure 5: Plot laugh2.wav

### 2.3 Longer explanation

The convolution operation matches the two inputs and do calculation. The output size is $len(signal) + len(impulse) - 1$. In our case, the convolution blends the original waveform with the impulse response and adds tail reflections.

## 3 Image filtering and enhancement

### 3.1

Figure 6 shows the original image as well as the image with salt and pepper noise and gaussian noise added.

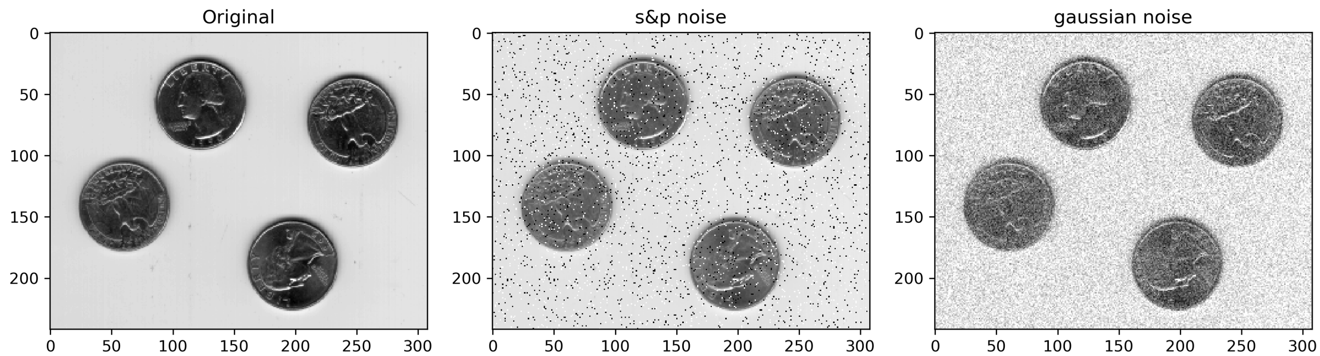



Figure 6: Original image and images with different types of noise

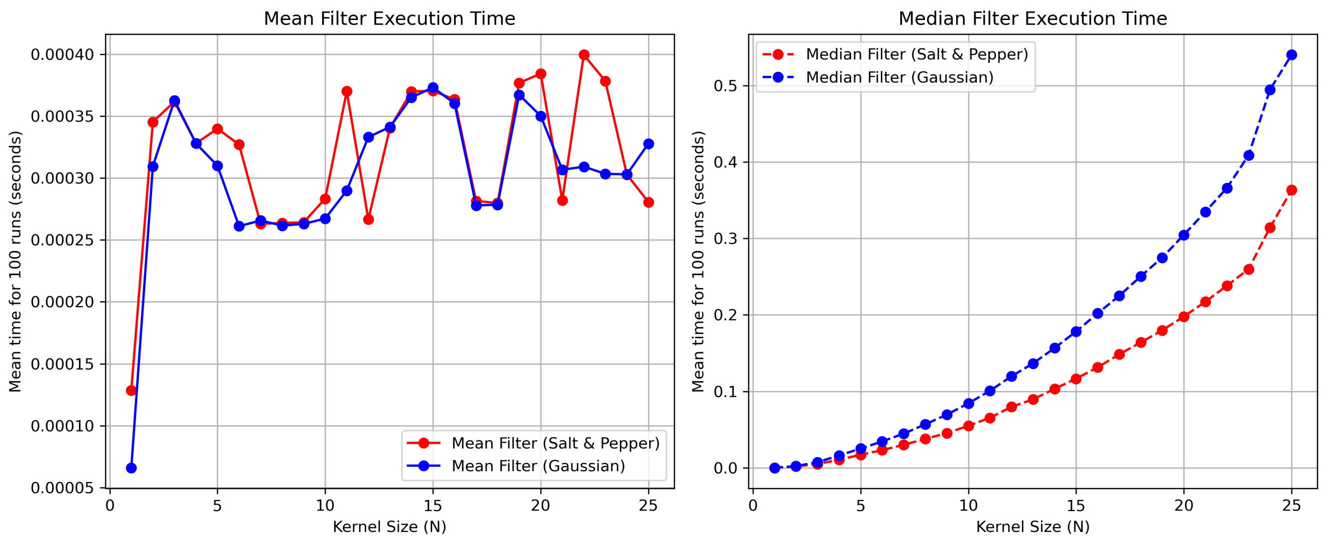



Figure 7: Different Average times obtained for N=1 to N=25 and each time for 100 executions

From figure 7, The computational time for Mean filter is low and varies less with kernel size, whereas the computational time for Median filter grows exponentially with kernel size and is particularly time-consuming for Gaussian noise.

For figure 8, As the kernel size N increases, the filtering effect increases and the image becomes blurrier. The effect of smoothing the noise by Mean filter is obvious, but it will lead to blurring of edges, especially when N = 25, the details are lost severely. The Median filter removes salt and pepper noise better and retains image edge detail better. Even though N increases, it still retains more detail than the mean filter.

## 3.2

From Figure 9, I think that when N exceeds 21, the visual effect of the image no longer changes significantly. I think the reason for this is that when N is large enough, the Gaussian kernel acts close to the entire image. At this point the image can also be considered as a large uniform blur. So even if N is increased, there is no significant change in the image.

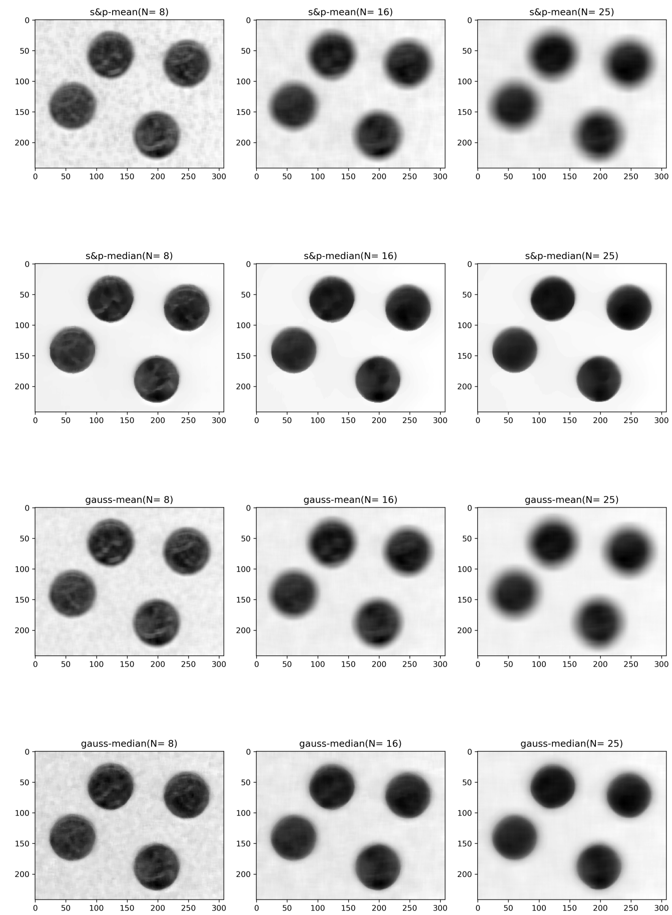



Figure 8: Visual effect of image with different types of noise by increasing the kernel size N

### 3.3

From figure 10, I found that Gaussian filter is suitable for removing Gaussian noise, not for salt and Pepper noise. Too little $\sigma$ is not enough for denoising and too much $\sigma$ causes blurring, That is, the image loses a lot of sharpness. I think $\sigma$ = 7 - 9 is recommended as the best balance point, But it is also depends on the level of noise.

## 4 Histogram-based processing

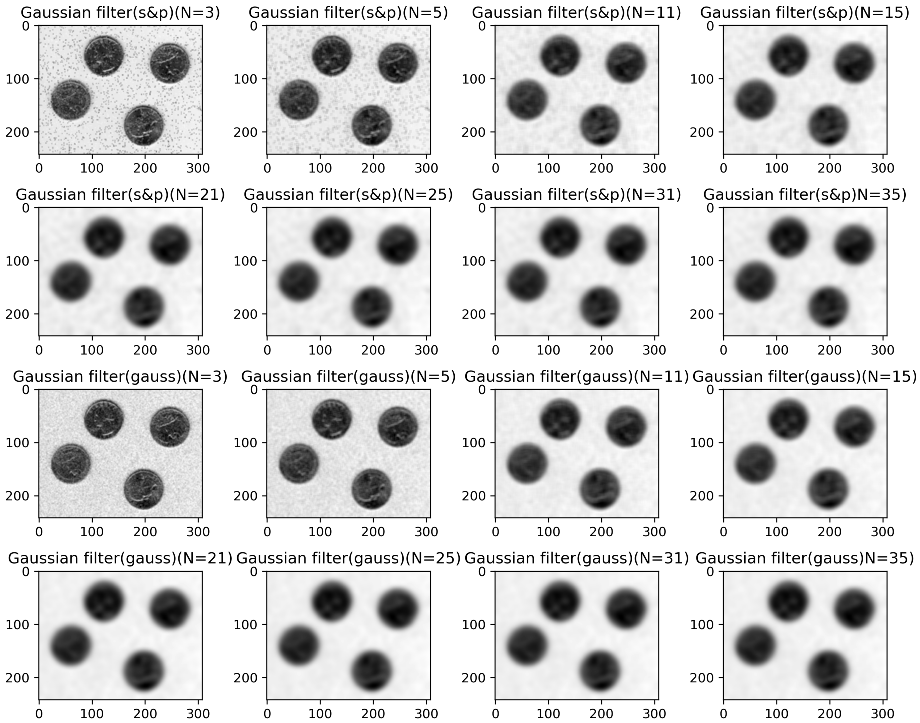



Figure 9: The effect of gaussian filter for different kernel size N(Sp and Gaussian noise)

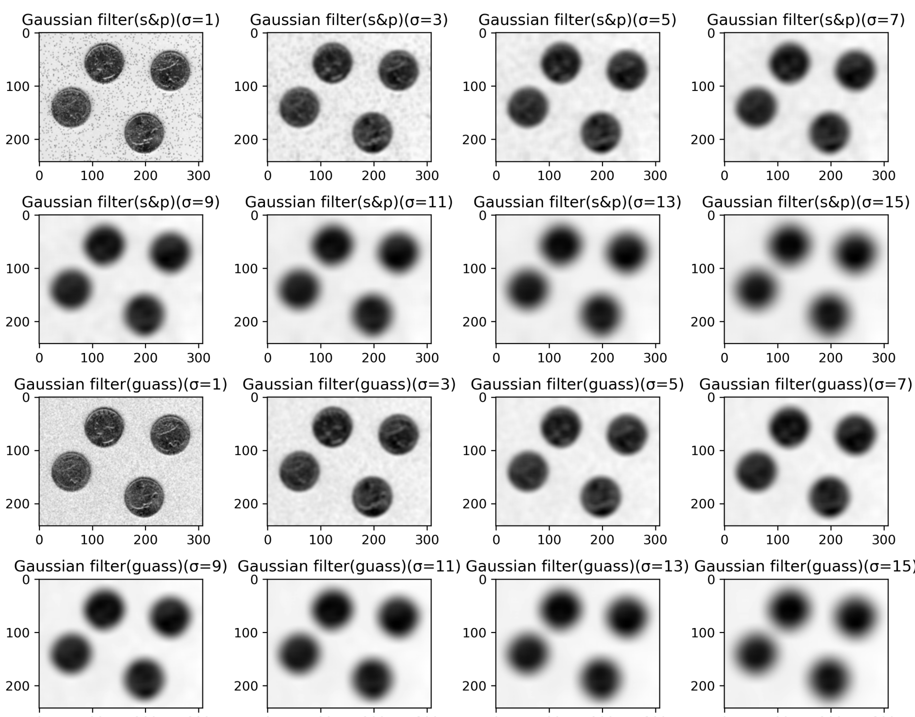



Figure 10: The effect of gaussian filter for different sigma(Sp and Gaussian noise)