EE5175: Image Signal Processing

Non-local means filtering

In this experiment, we will implement non-local means (NLM) filtering algorithm for the application of denoising.

You are given a noisy image, $\mathbf{g}$ (`krishna_0_001.png`), corresponding to a latent image, $\mathbf{f}$ (`krishna.png`), corrupted with additive Gaussian noise of mean 0 and variance 0.001. Your task is to apply NLM filtering on $\mathbf{g}$ following the steps in the given pseudocode to arrive at the denoised image, $\widehat{\mathbf{f}}$.

The parameters of the algorithm are the search neighbourhood radius W, the similarity neighbourhood radius W_{sim} and the filter parameter σ_{NLM}. A radius of W at a pixel denotes a window size of $(2W + 1) \times (2W + 1)$ around that pixel. The same applies to W_{sim}.

Q1 Show plots between the PSNR between $\mathbf{f}$ and $\widehat{\mathbf{f}}$ (y-axis) for different NLM filter parameter values $\sigma_{NLM} = 0.1$ to 1.0 in steps of 0.1 (x-axis) for the following search radius and similarity radius settings:
(a) $W = 5$, $W_{sim} = 3$,
(b) $W = 10$, $W_{sim} = 3$.
Show two plots in the same window with two different colours corresponding to (a) and (b). Compare the PSNR plots with the baseline PSNR between the noisy image $\mathbf{g}$ and the latent image $\mathbf{f}$.

Q2 We will now compare NLM filtering with the traditional Gaussian filtering. Denoise $\mathbf{g}$ using space-invariant Gaussian filter with $\sigma_g = 0.1$ to 1.0 in steps of 0.1 having a kernel window size of 11×11 for all σ_g values. Calculate the PSNR between the denoised images and $\mathbf{f}$. Add this plot to the plot window in Q1.

For the following filtering settings: (a) $W = 5$, $W_{sim} = 3, \sigma_{NLM} = 0.5$ for the NLM filtering, and (b) $\sigma_g = 1.0$ for Gaussian filtering, and at the following pixel locations $\mathbf{p}$: (i) row = 63, column = 93, and (ii) row = 77, column = 118, (total four combinations), do **Q3** and **Q4**.

Q3 Show the 11×11 filter (kernel) as an image. Comment.

Q4 Show the 11×11 image patch from the noisy image and the denoised images. Comment.
Use the `'InitialMagnification'` option in `imshow` command with a value greater than 100 (say, 1000) to display larger pixel sizes for patches and kernels.

Pseudocode

Read the noisy image $\mathbf{g}$ and the latent image $\mathbf{f}$ in the intensity range $[0, 1]$.
for every pixel position $\mathbf{p}$ **do**
// Obtain similarity neighbourhood around $\mathbf{p}$
Take the RGB patch $\mathcal{N}_p$ around $\mathbf{p}$ of radius W_{sim} in the image $\mathbf{g}$
Vectorize the patch $\mathcal{N}_p$ as a column vector $\mathbf{V}_p$

// Form the filter $\mathbf{w_p}$ at pixel $\mathbf{p}$.
// It can be formed as a 1D vector and visualized as a 2D matrix.
// We form a single filter for all three colour components.
for every pixel position $\mathbf{q}$ around $\mathbf{p}$ within radius W **do**
// Obtain similarity neighbourhood around $\mathbf{q}$
Take the RGB patch $\mathcal{N}_q$ around $\mathbf{q}$ of radius W_{sim} in the image $\mathbf{g}$
Vectorize the patch $\mathcal{N}_q$ as a column vector $\mathbf{V}_q$

The value of the filter $\mathbf{w_p}$ for the position $\mathbf{q}$ is given by

$$\mathbf{w_p}(\mathbf{q}) = \exp(-(\mathbf{V}_p - \mathbf{V}_q)^T(\mathbf{V}_p - \mathbf{V}_q)/\sigma^2_{NLM})$$

end for
Normalize $\mathbf{w_p} \leftarrow \mathbf{w_p}/\sum \mathbf{w_p}$

// Obtain search neighbourhood patch around $\mathbf{p}$
Take the RGB patches $\mathcal{N}_p^W(\mathtt{R})$, $\mathcal{N}_p^W(\mathtt{G})$, $\mathcal{N}_p^W(\mathtt{B})$ around $\mathbf{p}$ of radius W in the image $\mathbf{g}$ separately
Vectorize them as column vectors $\mathbf{V}_p^W(\mathtt{R})$, $\mathbf{V}_p^W(\mathtt{G})$, $\mathbf{V}_p^W(\mathtt{B})$

// Calculate the filtered output at pixel $\mathbf{p}$
// Use the same filter for all colour channels
The intensity at the output pixel $\mathbf{p}$ for each colour channel is given by

$$\widehat{\mathbf{f}}(\mathbf{p}, \mathtt{R}) = \mathbf{V}_p^W(\mathtt{R})^T \mathbf{w_p}$$
$$\widehat{\mathbf{f}}(\mathbf{p}, \mathtt{G}) = \mathbf{V}_p^W(\mathtt{G})^T \mathbf{w_p}$$
$$\widehat{\mathbf{f}}(\mathbf{p}, \mathtt{B}) = \mathbf{V}_p^W(\mathtt{B})^T \mathbf{w_p}$$

// Calculate the PSNR
// MSE : Mean Squared Error
// PSNR : Peak Signal-to-Noise Ratio
// The operation here assumes $\mathbf{f}$ and $\widehat{\mathbf{f}}$ are column vectors.
MSE = $(\mathbf{f} - \widehat{\mathbf{f}})^T(\mathbf{f} - \widehat{\mathbf{f}})$ / (total number of pixels including all colour channels)
PSNR = 10*log10(1 / MSE)
end for

Note:
1. For any two vectors, the inner product $\mathbf{a}^T\mathbf{b}$ calculates the element-wise multiplication followed by addition, and $\mathbf{a}^T\mathbf{a}$ results in the sum of the square of each element.
2. The general formula for PSNR is 10*log10(MAX*MAX / MSE), where MAX is the maximum image intensity value. We use MAX=1 in this experiment.

–