Comparison between several denoising methods for image processing

Jessy Khafif

March 17, 2023

Summary

Non Local Means (NLM)

Principle

- Iterative method
- Non Linear Filtering
- Using knowledge about neighbors

Filter Expression

$$w(i,j,k,l) = -e^{\frac{\|I(i,j)-I(k,l)\|_2^2}{2\sigma^2}}$$

$$I_D(i,j) = \frac{\sum_{k,l} I(k,l)w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)}$$

Filter Expression

$$w(i,j,k,l) = -e^{\frac{\|I(i,j)-I(k,l)\|_2^2}{2\sigma^2}}$$

$$I_D(i,j) = \frac{\sum_{k,l} I(k,l)w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)}$$

Algorithm

Algorithm 1 Filtering Algorithm

procedure DENOISING WITH NON LOCAL MEANS FILTER

Input: I, σ, (n_w, n_h)

Output: I_D

    for $pixel \in I$ **do**

        $neighs = neighboors\_of(pixel, n_w, n_h)$

        $I_D[pixel] = non\_local\_mean(pixel, neighs, \sigma)$

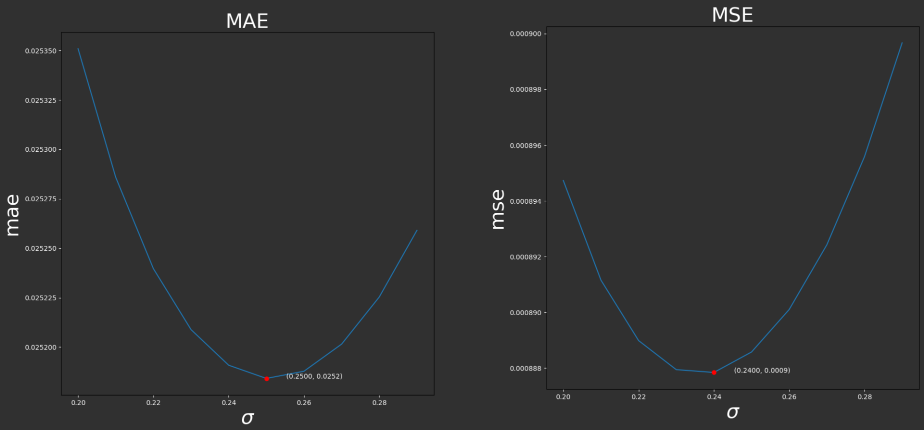
    end for

end procedure

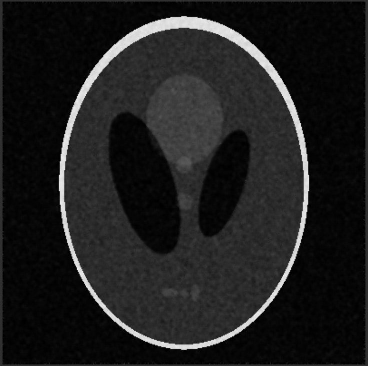
Data

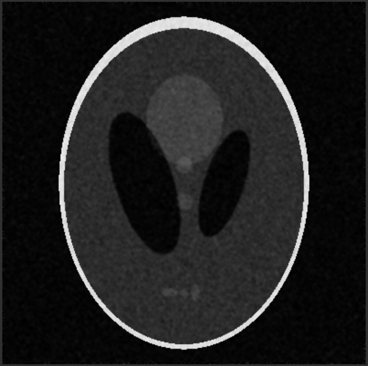

Noised (SNR_{db}= 10, MAE = 0.04390607221745121, MSE = 0.004277330275805813)

Results (with $(n_w, n_h) = (5, 5)$)



Results (with $(n_w, n_h) = (5, 5)$)





Bilateral Filter

Principle

- Iterative method
- Non Linear Filtering
- Using knowledge about neighbors
- Like NLM but we add an hyper-parameter related to the distance between pixels

Filter Expression

$$w(i,j,k,l) = e^{-\frac{(i-k)^2+(j-l)^2}{2\sigma_{spatial}^2} - \frac{\|I(i,j)-I(k,l)\|_2^2}{2\sigma_{color}^2}}$$

$$I_D(i,j) = \frac{\sum_{k,l} I(k,l)w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)}$$

Filter Expression

$$w(i,j,k,l) = e^{-\frac{(i-k)^2+(j-l)^2}{2\sigma_{spatial}^2} - \frac{\|I(i,j)-I(k,l)\|_2^2}{2\sigma_{color}^2}}$$

$$I_D(i,j) = \frac{\sum_{k,l} I(k,l)w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)}$$

Algorithm

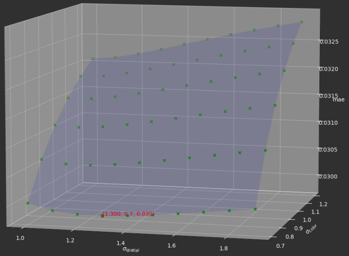
Algorithm 2 Filtering Algorithm

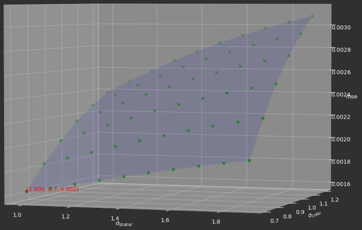
```
procedure DENOISING WITH BILATERAL FILTER
Input: I, σ_spatial, σ_color, (n_w, n_h)
Output: I_D
    for pixel ∈ I do
        neighs = neighboors_of(pixel, n_w, n_h)
        I_D[pixel] = bilateral_filter(pixel, neighs, σ_spatial, σ_color)
    end for
end procedure
```

Data

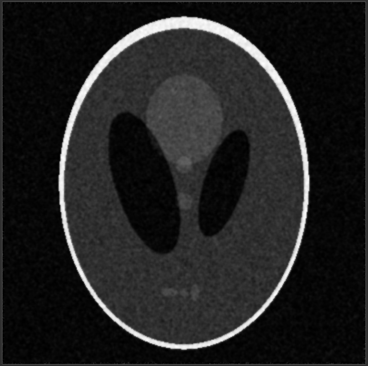
Noised (SNR_{db}= 10, MAE = 0.04390607221745121, MSE = 0.004277330275805813)

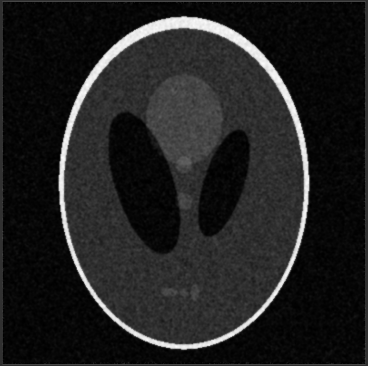
Results (with $(n_w, n_h) = (5, 5)$)





Results (with $(n_w, n_h) = (5, 5)$)





Anisotropic Filtering

Principle

- Iterative method
- PDE-based method

PDE meaning ?

Partial Differential Equation (PDE): Differential equation with a function as solution

Perona-Malik Model: Heat PDE

$$\begin{cases} I_0 = I_{\text{noisy}} \\ I_{k+1} = I_k + \lambda[\sum_{d \in Dir} (f_{\text{diffusion}} \circ (I_k * \nabla_d))] \end{cases}$$

- $\lambda =$ hyper-parameter
- $Dir = \{\text{North}, \text{East}, \text{South}, \text{West}\}$
- $k =$ iteration number
- $\nabla_d =$ derivation kernel with direction d
- $f_{\text{diffusion}} =$ heat diffusion function

Perona-Malik Model: Heat PDE

$$\begin{cases} I_0 = I_{\text{noisy}} \\ I_{k+1} = I_k + \lambda[\sum_{d \in Dir} (f_{\text{diffusion}} \circ (I_k * \nabla_d))] \end{cases}$$

- $\lambda =$ hyper-parameter
- $Dir = \{\text{North}, \text{East}, \text{South}, \text{West}\}$
- $k =$ iteration number
- $\nabla_d =$ derivation kernel with direction d
- $f_{\text{diffusion}} =$ heat diffusion function

Derivation kernels

$$\nabla_{\text{North}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\nabla_{\text{West}} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\nabla_{\text{South}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\nabla_{\text{Est}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Heat diffusion functions

$$f_{\text{diffusion}} : \mathcal{R}_+ \to \mathcal{R}_+^* \text{ such that } \begin{cases} f_{\text{diffusion}}(0) = 1 \\ \lim\limits_{u \to +\infty} f_{\text{diffusion}}(u) = 0 \end{cases}$$

Examples:

$$f_{\text{diffusion}}(u) = \frac{1}{1 + (\frac{u}{k})^2} \qquad f_{\text{diffusion}}(u) = e^{-(\frac{u}{k})^2}$$

with $k \in \mathcal{R}_+^*$

Heat diffusion functions

$$f_{\text{diffusion}} : \mathcal{R}_+ \to \mathcal{R}_+^* \text{ such that } \begin{cases} f_{\text{diffusion}}(0) = 1 \\ \lim_{u \to +\infty} f_{\text{diffusion}}(u) = 0 \end{cases}$$

Examples:

$$f_{\text{diffusion}}(u) = \frac{1}{1 + (\frac{u}{k})^2} \qquad f_{\text{diffusion}}(u) = e^{-(\frac{u}{k})^2}$$

with $k \in \mathcal{R}_+^*$

Notation about $f_{\text{diffusion}} \circ (I_k * \nabla_d)$

Set $M = I_k * \nabla_d$.

$$
\begin{aligned}
f_{\text{diffusion}} \circ M &= f_{\text{diffusion}} \circ \begin{bmatrix} m_{0,0} & \cdots & m_{0,M} \\ \vdots & \ddots & \vdots \\ m_{0,N} & \cdots & m_{M,N} \end{bmatrix} \\
&= \begin{bmatrix} f_{\text{diffusion}}(m_{0,0}) & \cdots & f_{\text{diffusion}}(m_{0,M}) \\ \vdots & \ddots & \vdots \\ f_{\text{diffusion}}(m_{0,N}) & \cdots & f_{\text{diffusion}}(m_{M,N}) \end{bmatrix}
\end{aligned}
$$

Algorithm

Algorithm 3 Filtering Algorithm

```
procedure DENOISING WITH ANISOTROPIC FILTER
Input: I, λ, N
Output: I_{N-1}
    I_0 = I
    for k ∈ [0, N[ do
        I_{k+1} = I_k + λ[ Σ_{d∈Dir} (f_diffusion ∘ (I_k * ∇_d))]
    end for
end procedure
```

procedure DENOISING WITH ANISOTROPIC FILTER

Input: I, λ, N

Output: I_{N-1}

$I_0 = I$

for $k \in [0, N[$ **do**

$$I_{k+1} = I_k + \lambda[\sum_{d \in Dir} (f_{\text{diffusion}} \circ (I_k * \nabla_d))]$$

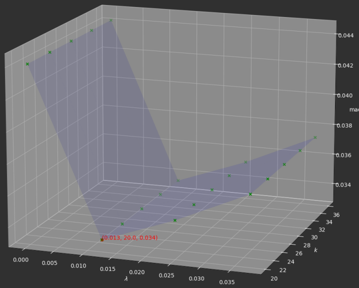
end for

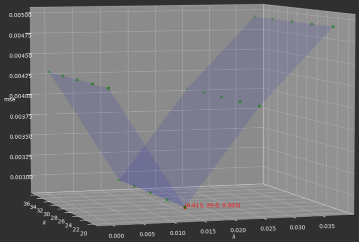
end procedure

Data

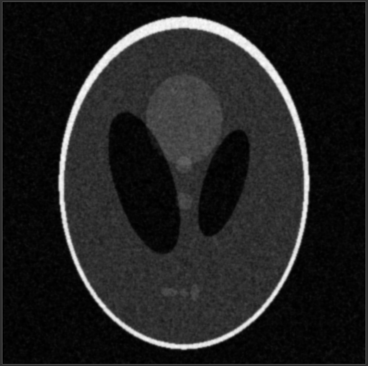
Noised (SNR_{db}= 10, MAE = 0.04390607221745121, MSE = 0.004277330275805813)

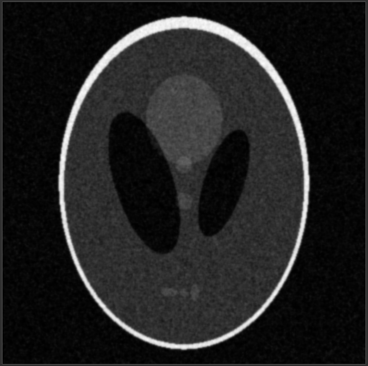
Results (with $N = 40$)





Results (with $N = 40$)





The End