Blockwise Nonlocal Means Denoising Filter for 2-D Images

Zhengran ZHU
Electrical and Computer Engineering
Concordia University
email: barodaslife@gmail.com

***Abstract*— A very serious issue in medical image processing is to remove the noise without losing any integrity of the original images, that is the image itself does not contort or lose any information due to the denoising process. Another important issue is that we want the algorithm to be executed as fast as possible without any delay, since the real-time surgery has been more widely implemented. This report is written to demonstrate that the newly proposed blockwise Non-local Means Filter can be 3 to 4 times faster in processing speed in 2-D domain and possibly 8 to 9 times faster in 3-D domain. A conventional processing algorithm which is called Non-local Means Filter is discussed in section 2 while the overall algorithm of Blockwise Non-local means filter will be discussed in section 3. And in the last section we compare these two algorithm together and multiple examples and simulations are given.**

I. INTRODUCTION

The conventional non local means filter requires that we check pixel by pixel throughout the entire image to calculate the weight that is completed by a Gaussian Kernel. In general we average the intensities of all the pixels in the image, and replace the pixel we intend to denoising. In practice, we only choose a neighborhood and check the similarity inside this neighborhood. However, this conventional non-local means filter takes a huge amount of time for the calculation which is not practical enough for the implementation especially for the real-time surgery in medical situations. When the situation become more complicated, for instance, in 3-D domain its calculation complexity becomes enormously large. No need to say that when other image processing tools are added such as image processing workflows (registration, segmentation, visualization, etc.), the image processing requires the denoising process in ahead to make more efficiency. Another interest implementation of non-local means is that it can help to restore image in some way or to help remove part of image naturally, which is a great help in all the image processing scenario.

Blockwise non-local means, instead of checking pixel by pixel, it loop inside the neighborhood block by block to check their similarity. We skip the pixels that are overlapping within the blocks so that the computation time is lesser, which gives more possibility to the real-time image processing. Moreover, we use another feature that is automatic tuning of the smoothing parameter in Gaussian Kernel so that the filter itself can become more adaptive and fully automated to overcome the main limitation of the classical Non-local means filter, that is the computational burden.

II. CONVENTIONAL NL-MEANS FILTER

A. Introduction

In most denoising methods, they restore the intensity value of each image pixel by averaging the intensities of its neighboring pixels in some way. In our case, the mostly used filter is the Gaussian filter, which weighing each pixel inside the neighborhood by its distance to the concerning pixel.

Such image processing technique can be called data-dependent approaches, which in aim of removing the neighboring pixels or voxels that is dissimilar to the concerning pixel or voxel. The most easiest way is the median filter that it can simply generalize the neighborhood pixels. Other sophisticated approaches, based on image derivatives have been successfully proposed for many applications, such as adaptive smoothing and AD. Generally speaking, all the existing techniques rely on the approach that we restore or substitute the concerning pixel depending on the similarity of the neighboring pixels.

B. Algorithm

To simplify our illustration, following notations are been used:

$u(x_i)$: Image where Ω^3 represents the grid of the image, considered as cubic for the sake of simplicity and without loss of generality($|\Omega^3| = N^3$)

For the original pixel-wise NL-Means approach:

$u(x_i)$	Intensity ovserved at voxel x_i.
V_i	square search volume centered on pixel x_i of size $\|V_i\| = (2M+1)^3, M \in N$
N_i	Cubic local neighborhood of $x_i$of size $\|N_i\| = (2d+1)^3, d \in N$
$u(N_i) = (u^(1)(N_i), ...,$ $u^(\|N_i\|)(N_i))^T$	Vector containing the intensities of N_i.
$NL(u)(x_i)$	Restored value of pixel x_i.
$w(x_i,x_j)$	weight of pixe; x_j when restoring $u(x_i)$

In the conventional NL-means filter, the restored intensity $NL(u)(x_i)$ of the pixelx_i, is the weighted average of all the pixel intensities in the image$u$ defined as

$$NL(u)(x_i) = \sum_{x_j \in \Omega^3} w(x_i,x_j)u(x_j) \tag{1}$$

where $u(x_j)$ is the intensity of pixel x_j and $w(x_i,x_j)$ is the weight assigned to $u(x_j)$ in the restoration of pixel x_i. To be more specific, the weight quantifies the similarity of the local neighborhood N_i and N_j of the pixel x_i and x_j under the assumptions that $w(x_i,x_j) \in [0,1]$ and $\sum_{x_j \in \Omega^3} w(x_i,x_j) = 1$. The original definition of Non-local means filter requires that weight has to be calculated through out the entire image. However, in practical, we limit the scope as so called "search volume" V_i of size $(2M+1)^3$, which is centered on the concerning pixel, to reduce the computational complexity. For each pixel inside search volumn V_i, the Gaussian-weighted Euclidean distance $||.||^2_{2,a}$ is computed between $u(N_j)$ and $u(N_i)$. This distance is a classical L_2 norm convolved with a Gaussian kernel of standard deviation a to measure the distance between neighborhood intensities. $w(x_i,x_j)$ is computed as follows:

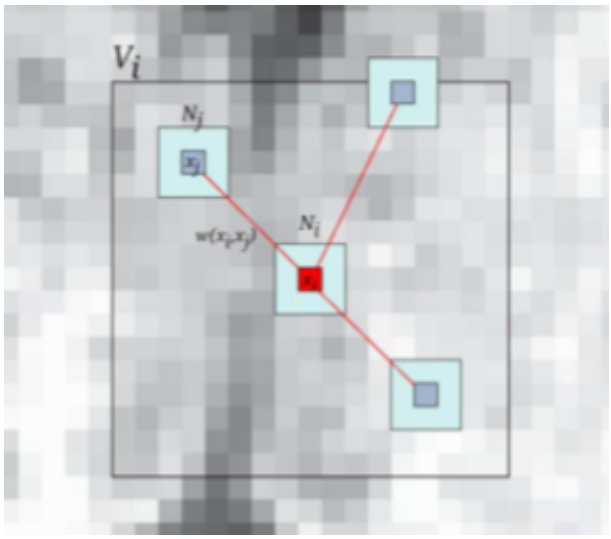

fig.1 Conventional NL Means Filter.

$$w(x_i,x_j) = \frac{1}{Z_i} e^{-\frac{||u(N_i)-u(N_j)||^2_{2,a}}{h^2}} \tag{2}$$

where Z_i is a normalization constant to ensure that $\sum_j w(x_i,x_j) = 1$, and h as a smoothing parameter controlling the decay of the exponential function. When h is high, all the pixel x_j in V_i will have the same weight $w(x_i,x_j)$ with respect to the pixel x_i. In this case, the restored value will be highly possible to be the average of the intensity values of the pixels in V_i, that is a strong smoothing of the original image. On the contrary, if h is very low, fewer pixels x_j in volume V_j will be similar to x_i. The restored value will tend to be the weighted average of those pixels with a similar neighborhood to concerning pixel x_i. However, this non-local means filter has its limitations, that is, in large images or even 3-D images, it will take significant long time to process, which give the minor possibility in real-time surgery or prepare for the following steps of image processing that requires instant output of denoising.

III. Blockwise Non-local Means Filter

In this section, the algorithm of a blockwise non-local means filter has been proposed, this method has been found much faster than the conventional non-local means filter due to its less computational complexity. The algorithm is consisted of 3 steps: 1.divide the search volume into several blocks. 2. Follow the same steps in convolution of the conventional non-local means filter. 3. restore the pixel values from those blocks that are centered on the concerning pixel.

To simplify the illustration, we use the following notation:

B_i	block centered on x_i of size $\|B_i\| = (2\alpha+1)^3, \alpha \in N$
$u(B_i)$	Vector containing the intensities of block B_i
$NL(u)(B_i)$	Vector containing the restored value of B_i
$w(B_i,B_j)$	Vector containing the intensities of N_i
$NL(u)(x_i)$	Restored value of pixel x_i
$w(x_i,x_j)$	Weight of B_j when restoring the block $u(B_i)$
B_{i_k}	blocks centered on pixel x_{i_k}.

To be more specific:

1. we choose 3 or 4 blocks around concerning pixel to make sure chosed blocks are all centered on the concerning pixel. The block size is of $(2\alpha+1)^3$. To ensure a global continuity in the denoised image, the support overlapping of blocks has to be nonempty: $2\alpha \geq n$.

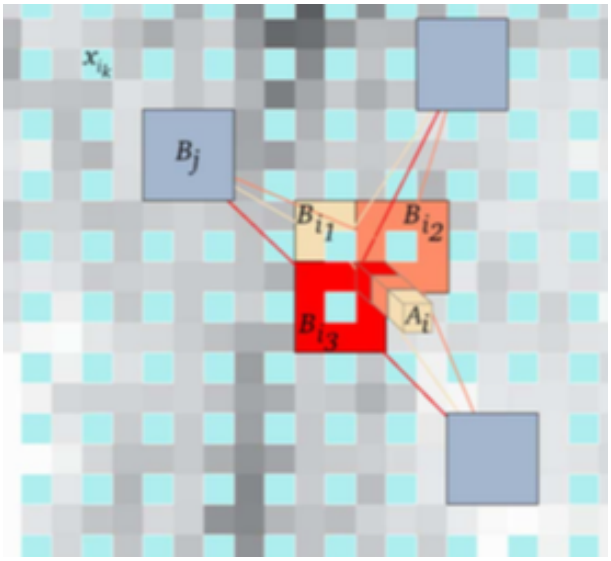

fig.2 Blockwise NL Means Filter.

2. For each block B_{i_k}, we perform the conventional way of non-local means filter to loop all the blocks inside the search volume. Depending on the block size, we skip several pixels each time. This is they way we reduce computational complexity:

$$NL(u)(B_{i_k}) = \sum_{B_j \in V_{i_k}} w(B_{i_k},B_j) u(B_j) \tag{3}$$

with:

$$w(B_{i_k}, B_j) = \frac{1}{Z_{i_k}} e^{-\frac{||u(B_{i_k}) - u(B-j)||_2^2}{h^2}} \tag{4}$$

where Z_{i_k} is a normalization constant ensuring that $\sum_{B_j \in V_{i_k}} w(B_{i_k}, B_j) = 1$

3. For a pixel that included in several blocks B_{i_k}. We do all the estimation about a total of selected blocks which all centered on concerning pixel to obtain $NL(u)(B_{i_k})$. After that, all the estimation of all the blocks are stored in a new vector A_i. The final restored intensity of pixel x_i is then defined as:

$$NL(u)(x_i) = \frac{1}{|A_i|} \sum_{p \in A_i} A_i(p) \tag{5}$$

That been said previously, this new method has the advantage of significantly reducing the computational complexity of the image. To be more specific, the conventional non-local means filter requires checking the similarity pixel by pixel throughout the search volume. However in this new method, we skip certain pixel that overlapping with the blocks. The bigger the block size, the faster the algorithm is. However, image is likely to reduce integrity if the block size is too big. This can be an interesting future work to identify the difference that the block size would make in particular scenario.

IV. Simulation Results

A. NL-means Filter

For conventional local means filter. We define the Gaussian Kernel to be $5 * 5$, degree of filtering to be 10 which is h. The general realization of coding in matlab consist of following several steps:

```
define a Gaussian Kernel 5*5
pad the image by 2
        for ij = (image size)
            define neighborhood
            define the search volume centered on x_i
                for mn = volume size
                    wij = exp(-norm/sigma^2)
                    NL1 = sum(wij*u(Xik)
                end
            NL(Xi)=average(NL1)
        end
```

show image To visualize more specificly. we use matlab to add noise into image so that we could use any image we used. The noisy image can be shown as below:

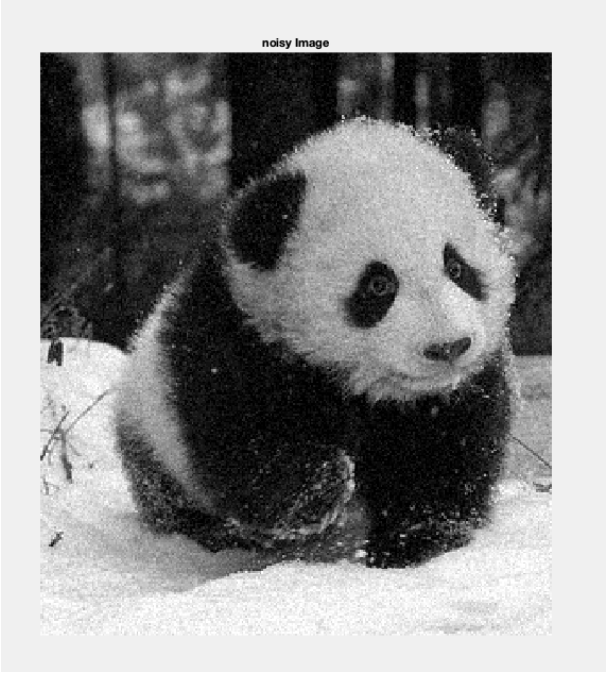

Fig.3 Noisy image

After been denoised with Non-local Means filter. The image after denoising is showed as follows:

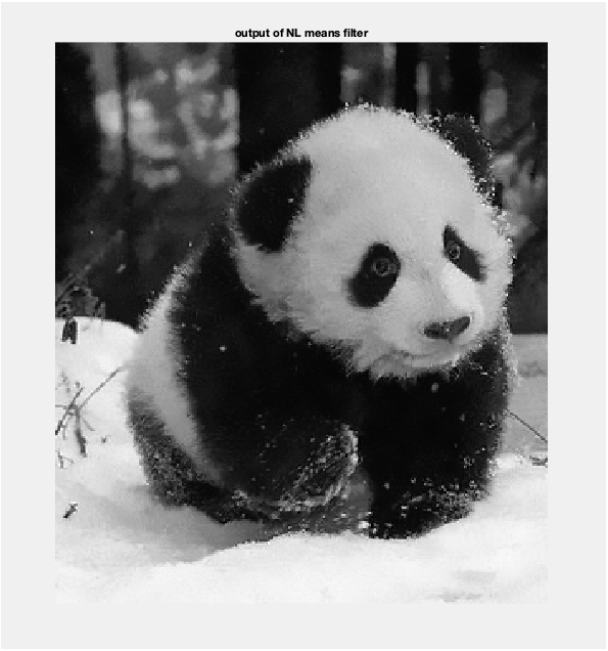

Fig.4 Denoised image by NL means filter

B. Blockwise NL-means Filter

For the Blockwise NL-means Filter, we use the following pseudocode:

```
define a Gaussian Kernel 5*5
pad the image by 2
        for ij = (image size)
            define reference pixels(center of 4 blocks)
            define the volume volumes of 4 blocks
                for mn = volume size
```

```
                    wij = exp(-norm/sigma^2)
                    NL1 = sum(wij*u(Bik)
                end
                Ai = NL1,2,3,4
            NL(Xi)=average(Ai)
        end
    show image
```

We denoise the same image shown above and get the results as follows.

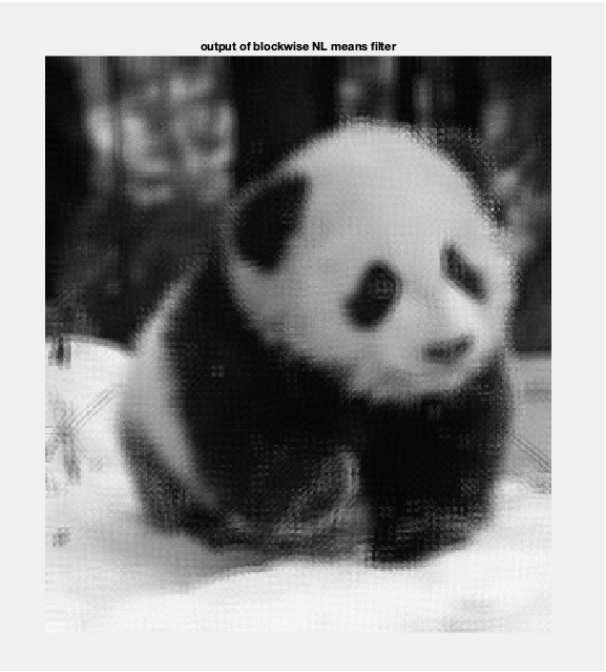

Fig.5 Denoised image by NL means filter.

As we can see from this output of denoising. The image is becoming blurring and less detailed. The reason of this outcome could be a very interesting future work. As for a assumption here, we think it is mostlikely because we jump or skip too many pixels when using the Gaussian kernel to do the convolution, since comparingly in the conventional NL-means filter, it check the similarity pixel by pixel inside the volume. In order to solve this problem, we tried to smaller the size of Gaussian kernel, which make sure there are more pixels been counted. Insteand of using $5*5$ Gaussian kernel, this time, we use a $3*3$ Gaussian kernel. We have our results again in the following:

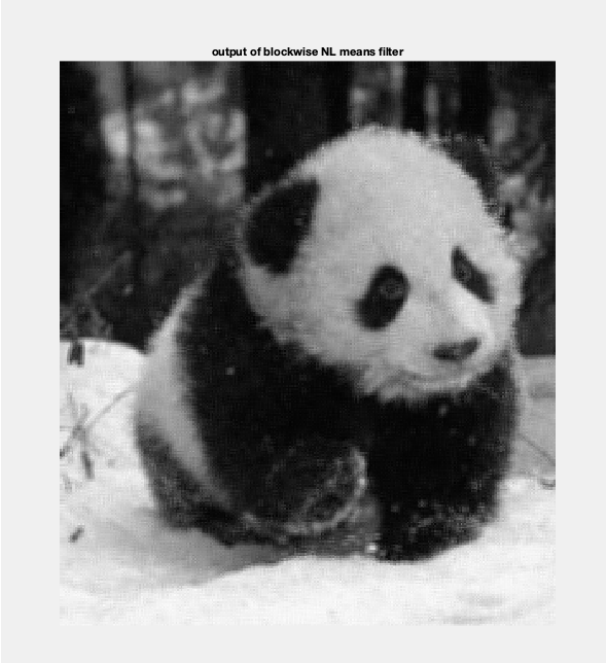

Fig.6 Denoised image by NL means filter(3*3).

We see that the now result is much better then the previous one. The denoised image is more clear and less blurry. And the noisy has been controlled without any compromise as well. Another thing that worth attention is that among this two gaussian kernel, they use almost the same time to process the image. One assumption for this interesting phenomenon is that, since we reduced the size of gaussian kernel, it not only increase the number of pixels been convoluted, but also decrease the computational complexity of the convolution itself. When we denoise the image using the conventional NL-menas filter, it took us 24.563 seconds to process the algorithm, while using Blockwise NL-means filter, it is much faster, only 8.754 seconds. It is 3 times faster then the conventional way, and possiblly 8 to 9 times faster in 3-D domain, although we do not get the result in the same quality. Future work would be some full comparision of thess two method, upon if maintain the same outcome quality, how fast can Blockwise NL-means go.

C. Difficulties Encountered

The main problem is for the realization of both NL-means and Blockwise NL-means filter is that in the actual code, the pixels on the edges and corners are difficult to be defined search volume on as well as the neighborhood, since it will go beyond the image. In our project, we use the pad image function in Matlab to pad those new pixels right beneath the boundary of images. In this case, we are able to process the pixels on the edges and corners. The below is how it works:

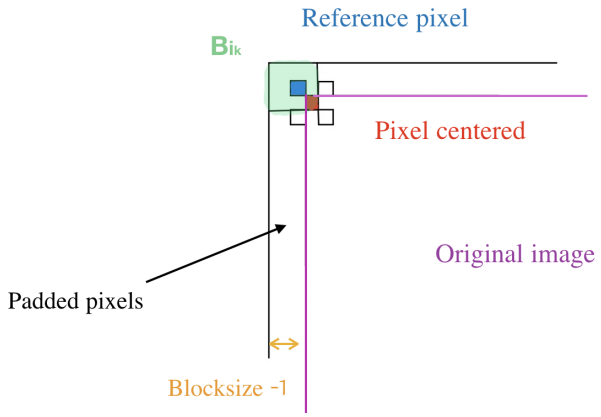

Fig.7 Padding images using pad function in matlab.

The padded pixels outside the boundary, its width is exactly enough for us to form the neighborhood or the search volume. The purple lines form the original image, and the black lines form the padded image. When coding we have to compensate off those pixels been padded.

V. Conclusion

In this report both the conventional NL-means filter and Blockwise NL-means filter are implemented in matlab code. we have done some comparision upon them though not thoroughly. The blockwise NL-means filter is clearly faster then the conventional filter, but its outcome is in low quality. In the original paper, Some optimised NL-means filter method had been proposed. They add more parameters into the gaussian kernel to make it adjustable for different situations. Also parallel computation is proposed in original paper as well to reduce the processing time. This can be done by matlab using parallel computation.

And finally in the simulation results section. We proposed our pseudocode to better illustrate our idea of implementing both NL and Blockwise NL means filter, some results are given. The proposed results show that in terms of overall efficiency, Blockwise NL-means filter is much better then the conventional NL-means filter.

References

[1] Pierrick Coup*, Pierre Yger, Sylvain Prima, Pierre Hellier, Charles Kervrann, and Christian Barillot, FAn Optimized Blockwise Nonlocal Means Denoising Filter for 3-D Magnetic Resonance Images, April, 2008

[2] GPierrick Coup,1Pierre Hellier,* Charles Kervrann,and Christian Barillot, Nonlocal means-based speckle filtering for ultrasound images.May 27,2009.