Implementation of Non-local mean denoising in OSC imaging

BIE Xiaoyu

Abstract

In the polarization imaging, the noninformative spatial fluctuation needs to be reduced in some case. One of the simplest way to achieve it is the orthogonal state contrast(OSC).[1] In this technique, we illuminate a target with a give polarization state and acquire two intensity images in parallel and orthogonal direction. The OSC image can be obtained by calculating the ratio between the difference and the sum of these two images. However, these images are frequently affected by different sources of noise, which leads to the decrease of contrast between the target and background. In this article, we introduce the non-local mean denoising algorithm into the OSC images and study its performance. This denoising algorithm can reduce the noise in the images while it can also reserve its resolution.

1 Introduction

1.1 OSC imaging system

An imaging system which measures the degree of polarization(DOP) is of great interest in detection domain.In a polarimetric system, we can obtain two images with different polarization direction. In some cases, light intensity is not very important or even unfavourable for image detection system because of the false alarms. In this situation, a special decomposition to suppress the influence of intensity is necessary.

The simplest way to solve it is the orthogonal state contrast(OSC). [1] In this method, we can acquire two intensity images by analyzing in parallel and orthogonal directions. The OSC image is obtained as the ratio of the difference and the sum of these two images:

$$P = \frac{X - Y}{X + Y}$$

where X and Y are the two intensity images, P is the OSC image and we also call it degree of polarization(DOP). This transformation can provide an image suppressing the influence of intensity, however, it can also enhance the noise because the image is a ratio of noisy measurement. This noise has already been studied and its probability density function is given as:[2]

$$P(\rho) = \alpha P_1(\rho) + (1 - \alpha) P_2(\rho)$$

where ρ is the ratio of two independent normal random variables with the same variance but different means and $\alpha = exp(-SNR^2(1 + P^2)/2)$. We also have:

$$P_1(\rho) = \frac{1}{\pi(1 + \rho^2)}$$

$$P_2(\rho) = \frac{SNR}{\sqrt{2\pi}(1 - \alpha)} \frac{1 + \rho P}{(1 + \rho^2)^{3/2}} F\left(SNR \frac{(1 + \rho P)}{\sqrt{1 + \rho^2}}\right) \times exp\left(-\frac{SNR^2(\rho - P)^2}{2(1 + \rho^2)}\right)$$

where $F(x) = [erf(x/\sqrt{2}) - erf(-x/\sqrt{2})]/2$ and $erf(x) = 1/\sqrt{\pi} \int_0^x exp(-t^2)dt$.

We can also define the Cramer-Rao Lower Bound(CRLB) on the estimation of the P:[2]

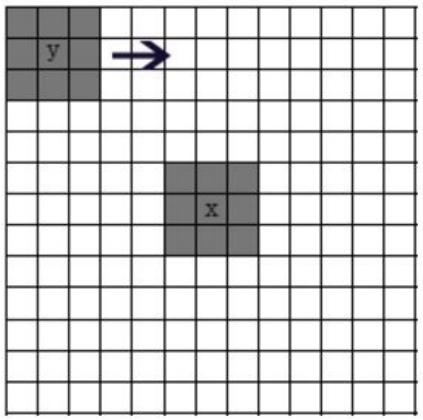
$$CRLB = \frac{1}{N} \frac{(1 + P^2)}{SNR^2}$$

where N is the time of intensity measurements.

1.2 Non-local mean denoising

The non-local means denoising was first presented by Antoni Buades et al. in 2005[3]. The principle of this method is to replace the pixel value with the average of nearby pixels. Different form the classic filtering algorithm, non-local means give the weight of nearby pixels on comparing the similarity of the structure between the two pixels.

In this method, we first define two windows in the image, one is called research window while the other is called patch window.



As shown in the left figure, the big window is the research window centered at x which is the pixel we want to denoise. In the research window, we build two patch windows with the same size. which are marked with gray in the figure. One is also centered at x while the other is centered at y, a certain pixel moving in the research window.
We build the research window to define the range of all the pixels nearby which participate in the calculation and the patch window is used to calculate the similarity.

Figure 1: Research window in non-local means denoising[1]

The restored image $\hat{u} = (\hat{u}_1, \hat{u}_2, \hat{u}_3)$ is an average of the most resembling pixels, where the resemblance is computed in a research window. A certain pixel p writes[4]:

$$\hat{u}_i(x) = \frac{1}{C(x)} \sum_{y \in B(x,Ds)} \tilde{u}_i(y) w(x,y), \qquad C(x) = \sum_{y \in B(x,Ds)} w(x,y)$$

where $i = 1, 2, 3$ which means the 3 dimensions in a RGB image, $B(x, Ds)$ indicates the research window centered at x and with a size of $(2Ds+1) \times (2Ds+1)$ and $w(x,y)$ is the weight between the pixel x and pixel y. The $C(x)$ is a normalized coefficient. Here and in the following equations, we consider $u(i), \tilde{u}(i)$ and $\hat{u}(i)$ as the original image value, the noisy image value and the restored image value.

1.3 Weight calculation

The weight of each pixel in the research window depend on its similarity of structures between x and y, which is called squared Euclidean distance:

$$d^2(x,y) = \frac{1}{3(2ds+1)^2} \sum_{i=1}^{3} \sum_{z \in B(0,ds)} |u_i(x+z) - u_i(y+z)|^2$$

where $B(0, ds)$ is a squared window with a size of $(2ds+1) \times (2ds+1)$ centered in $(0,0)$. Then, we use an exponential kernel to compute the weights:

$$w(x,y) = exp\left(-\frac{max(d^2 - 2\sigma^2, 0)}{h^2}\right)$$

where σ denotes the standard deviation of the noise and h is a filtering parameter depending on value of σ. Here, we use a max function to set the weight to 1 when the squared Euclidean distance is smaller than $2\sigma^2$ in order to avoid overweighting the pixel which has a large similarity. We also set the weight of reference pixel x to the maximum of the weights in the research window $B(x, Ds)$

1.4 Parameters selection

In order to obtain the better result, the size of the research window and patch window depend on the value of σ. With a larger σ, which means the image contains more noise, we need a larger research window to increase the noise removal capacity.

The value of the filtering parameter is $h = k\sigma$, where k will decrease when the size of patch window increase. The two patch window with a larger size will concentrate more around $2\sigma^2$ so that we need a smaller k.

These parameters are given below and were made by Antoni Buades al, and we will use them in the simulations in Section 2:

σ	Patch Window	Res. Window	h
$0 < \sigma \leq 15$	3×3	21×21	0.40σ
$15 < \sigma \leq 30$	5×5	21×21	0.40σ
$30 < \sigma \leq 45$	7×7	35×35	0.35σ
$45 < \sigma \leq 75$	9×9	35×35	0.35σ
$75 < \sigma \leq 100$	11×11	35×35	0.30σ

Table 1: Gray Image[3]

σ	Patch Window	Res. Window	h
$0 < \sigma \leq 25$	3×3	21×21	0.55σ
$25 < \sigma \leq 55$	5×5	35×35	0.40σ
$55 < \sigma \leq 100$	7×7	35×35	0.35σ

Table 2: RGB Image[3]

1.5 Edge problem

As we know in the previous work, for each pixel, we have to build a research window centered at itself to optimize it. However, for the pixels at the edge of the image, there is not enough pixel to build the research window, so we choose to move our window and guarantee that it will not exceed the original image.

Then, we consider about the patch window, we build a patch window for each pixel in the research window. So for the research window who locate at the edge of the image, there also exist the edge problem. On this occasion, we pad our image with a mirror reflection and with a size of ds, the radius or the patch window.

The sketch for the solutions in these two occasions are shown in Figure 2 and Figure 3:

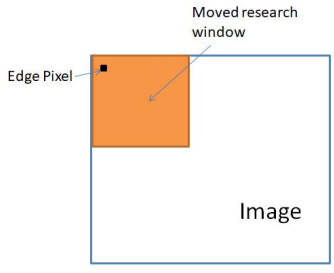


Figure 2: Solution for research window

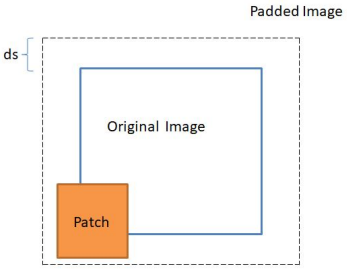


Figure 3: Solution for patch window

1.6 Evaluation

In order to evaluate the quality of the optimized image, we introduce the peak signal-to-noise ratio(PSNR). A higher value of PSNR means a higher quality of the image. This value is defined

as:

$$PSNR = 10log_{10}\left(\frac{MAX_I^2}{MSE}\right)$$

where the MAX_I is the maximum pixel value of the image(for a 8 bits image, it is 255) and the MSE is the mean square error, which is also the square of the Euclidean distance between the original image and the restored image.

2 Evaluation of the performance of NL-mean denoising

In order to test our algorithm, we choose a RBG image with a size of 469×704 and add a noise with $\sigma = 2$, we apply our algorithm in $Matlab$:

Figure 4: Implementation with a RBG image (a)Original image; (b)Noisy image with enlargement; (c)Restored image with enlargement

From the figure, we can easily see that the white part of the image with noise is well restored after the non-local means denoising. And we can also calculate its $\boxed{PSNR = 47.91}$, which means a pretty good result. However, the problem is its computing time. When applying the algorithm to this image in a notebook and in $Matlab$, the computing time is $\boxed{t = 1115s}$, and it is intolerable in practical application.

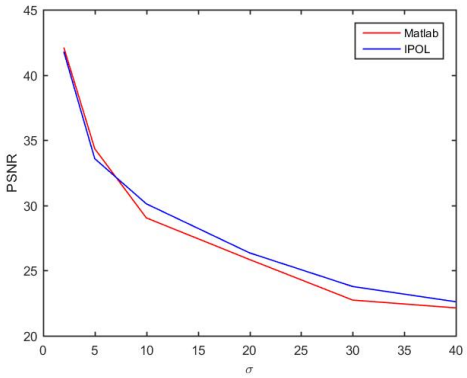
In the second step, we cut out a part of several images to reduce its computing time and just compare the result obtaining in $Matlab$ with the result obtaining in the $IPOL$, a website where the non-local means algorithm is published and it also gives an on-line computing for this method.

After implementation both in $Matlab$ and $IPOL$[5], we can obtain a table of PSNR:

We can plot the curve of PSNR on function of σ for the first figure:

where the red curve indicate the result obtained by $Matlab$ and the blue curve indicate the result obtained by $IPOL$. The results are pretty close and we can consider that the algorithm we used in $Matlab$ has the same performance as it is in the $IPOL$.

σ	2		5		10		20		30		40	
	Mat.	*IPOL*	*Mat.*	*IPOL*	*Mat.*	*IPOL*	*Mat.*	*IPOL*	*Mat.*	*IPOL*	*Mat.*	*IPOL*
P1	42.22	41.76	34.53	35.25	30.36	31.56	27.32	27.57	24.80	25.32	23.89	23.95
P2	45.13	45.39	40.62	41.01	36.63	37.13	33.01	33.34	30.47	30.83	29.00	29.38
P3	48.11	48.18	44.47	44.79	41.29	41.63	37.52	37.54	36.12	35.32	34.10	33.50
P4	42.75	42.82	36.36	37.46	32.27	33.82	28.89	29.48	25.29	26.97	24.66	25.17
P5	43.08	44.07	38.84	39.79	36.00	36.08	32.17	32.22	30.34	29.47	28.87	27.58
P6	42.23	42.89	36.61	37.39	32.58	33.71	29.68	29.87	26.95	27.58	26.05	26.24
P7	43.08	42.32	35.30	34.48	29.64	29.11	24.80	25.99	20.85	23.04	19.60	21.48
P8	42.11	41.79	34.34	33.58	29.04	30.12	25.85	26.36	22.74	23.78	22.14	22.61



2.1 Cartoon effect

In our previous analysis, the weight of a pixel in a research window depends on its Euclidean distance between the pixel we want to optimize and itself. This precess can denoise the noisy image but it will lose some details of the image and cause a cartoon effect:

Figure 5: Noisy image

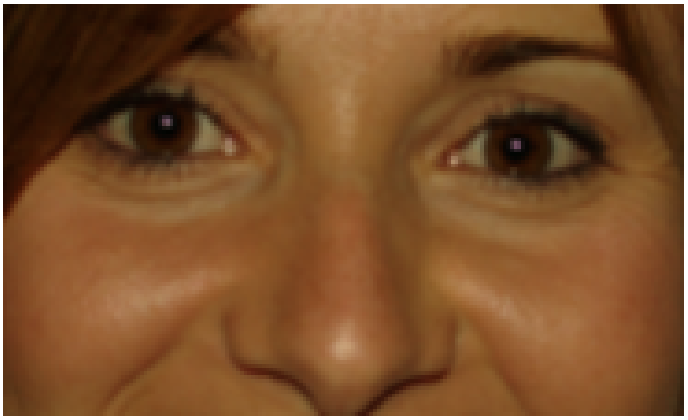


Figure 6: Original image

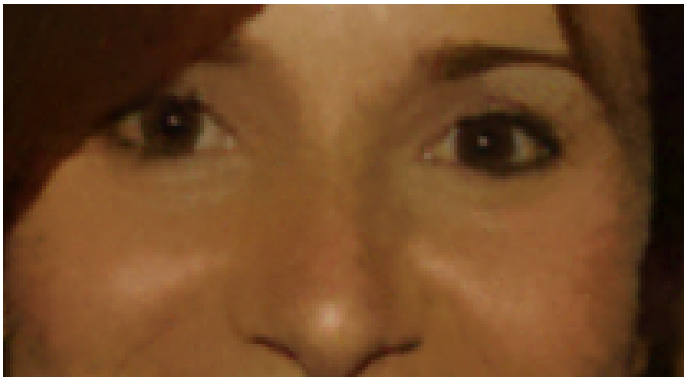


Figure 7: Restored image

In these figures, original image is noised with $\sigma = 20$, and restored image is well denoised with non-local means algorithm, however some details are lost, especially around the nose. The following work can concentrate on this effect and try to reduce it.

2.2 Computing time

As we know, the non-local means denoising introduce two types of window(research window and patch window), which means for optimizing each pixel we have to process a large number of calculation. It is why we cost near 1000s to optimize a 469×704 RGB image, and even for a part of it(100×100), it remains to cost around 30s.

3 Implementation of NL-means denoising algorithm in OSC images

In this section, we implement this denoising algorithm in OSC images. In order to better understand its performance, we firstly introduce it into a simple OSC images with gaussian noises to study the influence of parameters, and then we use a complex OSC image which is also noised with gaussian noises, so we can obtain its performance in ideal condition. Finally we implement this algorithm into several practical OSC images which contain some unknown noises,

3.1 Implementation in simple images with uniform background

3.1.1 Image implementation

In order to study the performance of non-local means algorithm in the denoising of the degree of polarization, we establish a simulation in *Matlab*. We choose 16 images with a size of 64×64 and their intensities are $I(i) = 7 + (i-1) \times 2.8667$,$i = 1, 2, ..., 16$(intensity is from 7 to 50), we also

add a small target in the center of image with different polarization performance and its size is of 4×4.

Then, we assume that the average of OSC is $P = 0.5$ and its difference is $\delta P = 0.2$. That means for the target we have $P_t = P + \delta P/2 = 0.6$ and for the background we have $P_b = P - \delta P/2 = 0.4$. Then we can obtain two images X and Y and deduce their intensities:

$$I_X(i) = I(i) \cdot \frac{1+P}{2}$$

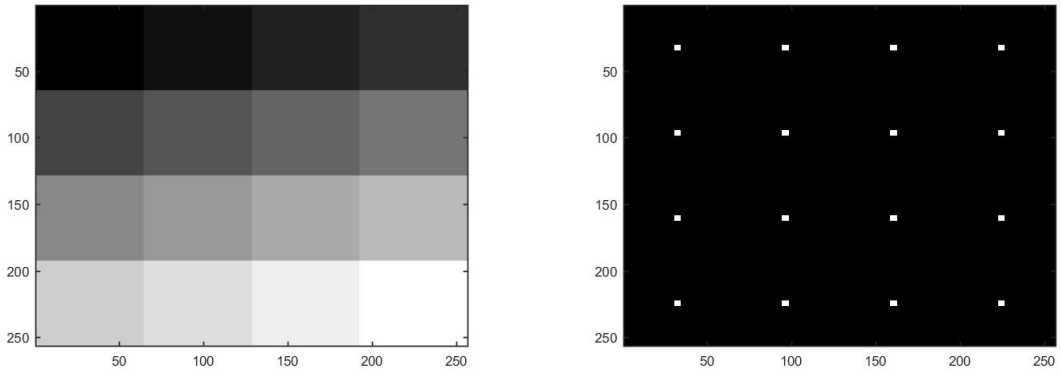
$$I_Y(i) = I(i) \cdot \frac{1-P}{2}$$

where P is P_t or P_b with respect to the target or background. And we can re-obtain the intensity of the image and the DOP :

$$I(i) = I_X(i) + I_Y(i)$$

$$P(i) = \frac{I_X(i) - I_Y(i)}{I_X(i) + I_Y(i)}$$

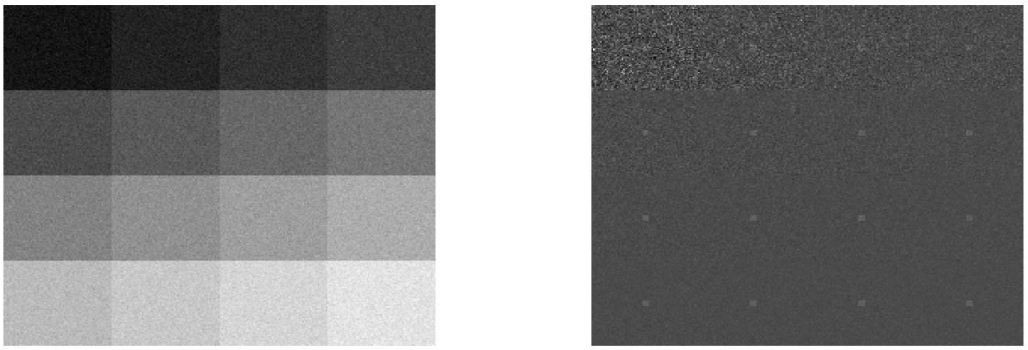
These images are shown with a format of 4×4 in Figure **??**:



(a) **(b)**

Figure 8: Original image (a) image of intensity, (b) image of DOP

Then, we add a noisy to these polarization images with certain variance $\boxed{\sigma = 1}$ and the results are shown in Figure 9:



(a) **(b)**

Figure 9: Noisy image (a) image of intensity, (b) image of DOP

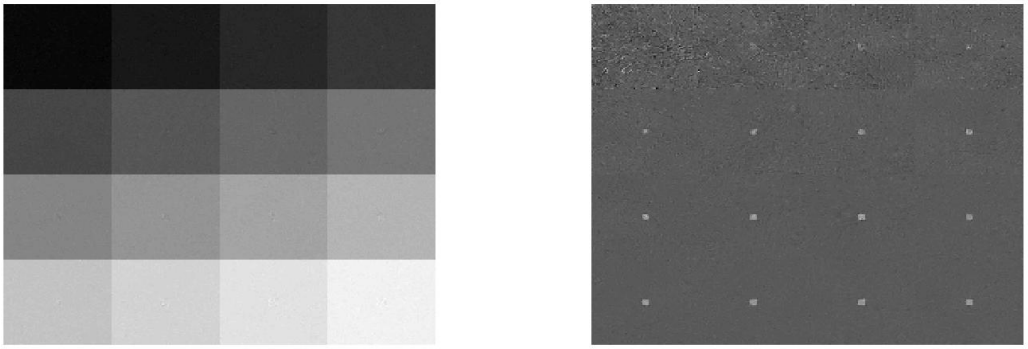
We can see that the DOP computed by the noisy polarization images has a very bad contrast, especially with a low intensity. Afterwards, we apply the non-local means denoising to the image X

and Y(I_X and I_Y), we choose the parameters in non-local means algorithm as $\boxed{ds = 1}$, $\boxed{Ds = 10}$ and $\boxed{h = 0.4\sigma}$ for the best optimization results[4], we note that the optimized images are X_new, Y_new and we can deduce their expressions of intensity:

$$I_{X\_new}(i) = \frac{1}{C_X(i)} \sum_{j \in B_X(i,Ds)} I_X(i) w_X(i,j) \qquad C_X(i) = \sum_{j \in B_X(i,Ds)} w_X(i,j)$$

$$I_{Y\_new}(i) = \frac{1}{C_Y(i)} \sum_{j \in B_Y(i,Ds)} I_Y(i) w_Y(i,j) \qquad C_Y(i) = \sum_{j \in B_Y(i,Ds)} w_Y(i,j)$$

and we can obtain the optimized intensity images and DOP images in Figure 10:



(a) **(b)**

Figure 10: Restored image (a) image of intensity, (b) image of DOP

In the following, we re-build the DOP figure by putting the noisy image and restored image together and ranking with intensity degree so as to better compare the performance of our filter:

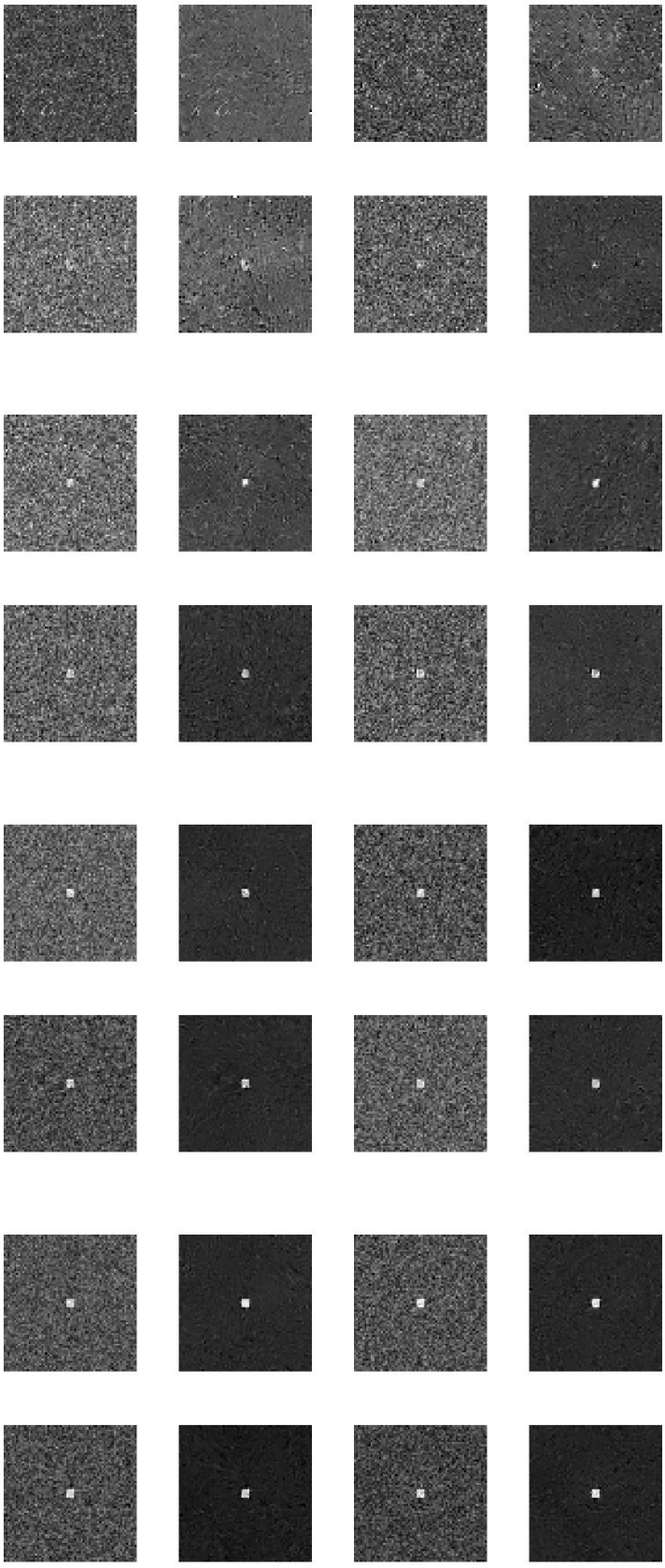


Figure 11: Comparison of DOP between noisy image and restored image, in order of intensity degree

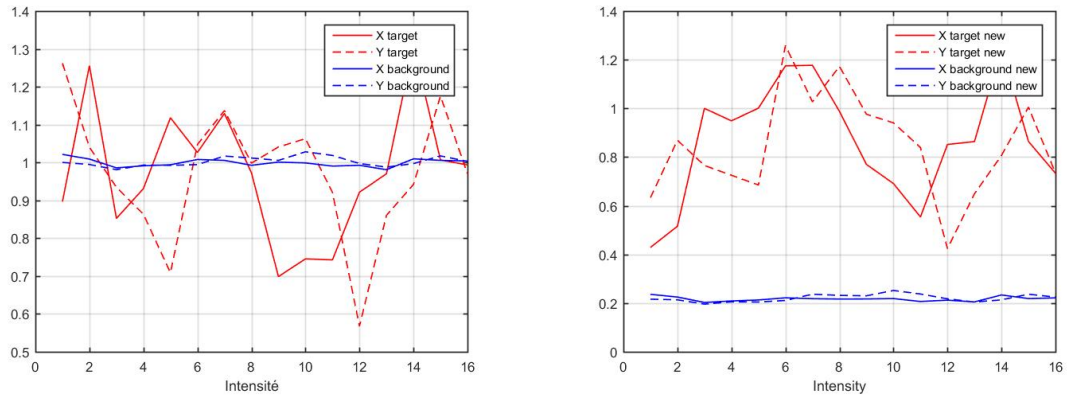
As shown in Figure 11, we can roughly deduce that in a low intensity, the non-local means denoising can reduce the noise of the DOP but the target is still hard to find out, while in a high intensity the noise can be decreased and the contrast between the target and background has increased.

3.1.2 Noise reduction in X and Y

In order to study the practical performance of the non-local means denoising in the DOP, we use the root mean square error(RMSE) to demonstrate the optimization results:

$$RMSE = \sqrt{\frac{1}{N}\sum(u'(i) - u(i))^2}$$

where $u'(i)$ is the noisy pixel $\tilde{u}(i)$ or restored pixel $\hat{u}(i)$, $u(i)$ is the original pixel and N is the number of pixels in the image. We plot the RMSE of image X,Y and image X_new,Y_new(the polarization image after denoising) in Figure **??**:



(a) **(b)**

Figure 12: RMSE (a) original polarization images, (b) restored polarization images

The axis X is the order of intensity degree i in the $I_X(i)$, $I_Y(i)$ and $I_{X\_new}(i)$, $I_{Y\_new}(i)$.The axis Y is the value of RMSE. The red curve and the blue curve indicate the target area and the background area, while the solid line denotes the image X and the dashed line denotes the image Y. Here, we find that the RMSE of the background seems to be equal to the value σ. This is reasonable because when we add noise to the image X and image Y in $Matlab$, we have:

$$\tilde{X}(i) = X(i) + X_b; \quad \tilde{Y}(i) = Y(i) + Y_b$$

where X_b and Y_b are gaussian noise with a means value of 0 and a standard deviation $\sigma = 1$, in the computing of RMSE, we have:

$$RMSE_X = \sqrt{\frac{1}{N}\sum(\tilde{X}(i) - X(i))^2} = \sqrt{\frac{1}{N}\sum(X_b)^2} \approx \sigma$$
$$RMSE_Y = \sqrt{\frac{1}{N}\sum(\tilde{Y}(i) - Y(i))^2} = \sqrt{\frac{1}{N}\sum(Y_b)^2} \approx \sigma$$

it is why the curve of RMSE of background is similar to σ, as for the RMSE of target, the number of samples is so small that the results are random.

Comparing the two figures in Figure **??**, we can deduce that, for the target area, the RMSE does not change too much, however for the background, the RMSE is reduced with around $\boxed{4.5 \text{ times}}$. We have to realize that for non-local means denoising in such degree of noise ($\sigma = 1$), the patch has a size of 3×3 which is close to the size of target and it has a research window of 21×21 which is much bigger than the size of target. This is why it seems to not work very well in denoising the target.

As for the part of background in the restored image, we suppose $b(i)$ is the noise in pixel i, for each image, the intensity is uniform and we denote it u so we have :

$$\begin{aligned}\hat{u}(i) &= \frac{1}{\sum w(i,j)} \sum w(i,j)\tilde{u}(j) \\ &= \frac{1}{\sum w(i,j)} \sum w(i,j)(u + b(j)) \\ &= u + \frac{\sum w(i,j)b(j)}{\sum w(i,j)}\end{aligned}$$

substitute :

$$\begin{aligned}RMSE(\hat{u})_{background} &= \sqrt{\frac{1}{N}\sum(\hat{u}(i) - u)^2} \\ &= \frac{1}{\sqrt{N}}\sqrt{\sum\left(\frac{\sum w(i,j)b(j)}{\sum w(i,j)}\right)^2}\end{aligned}$$

We can see that if $w(i,j)$ and $b(j)$ are independent, the RMSE in the restored image will be reduced by $1/\sqrt{N}$ times, which is around 20. However, the results in the simulation show that the RMSE only reduce by 4.5 times. This is because in the non-local means algorithm, the research windows for different pixels have overlapping parts, as well as the patch windows in the optimization of one pixel. Here, we define a ratio between the RMSE of noisy polarization images and the RMSE of restored images:

$$r = \frac{RMSE(\tilde{u})}{RMSE(\hat{u})}$$

where a higher ratio r means a better performance of the filtering in denoising. Obviously r reach the maximum value $\sqrt{N}$ when the it is a mean filter.

In order to study the influence of correlation in the performance of filtering, we choose a certain pixel (at (15,15), to avoid the effect of edge problem and the effect of target), Then we repeat to add noise and denoise so that the noise in each image are uncorrelated, we can repeat 1000 times to obtain its empirical main square error, comparing the noisy images and restored images, we can compute the ratio r:

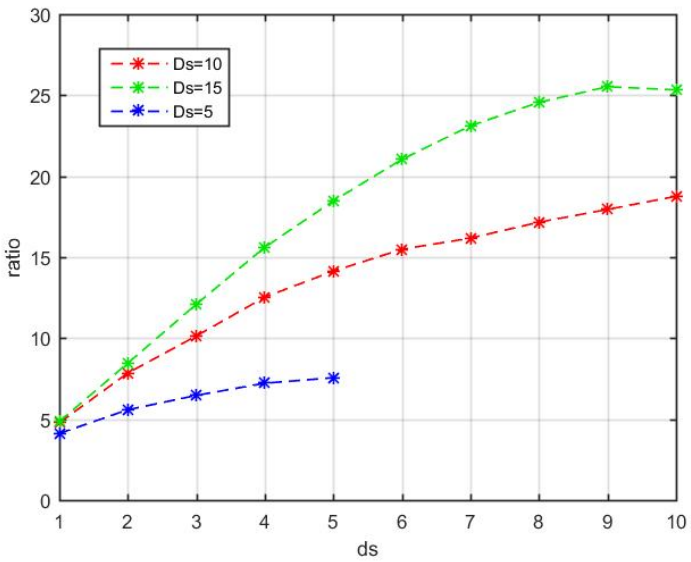
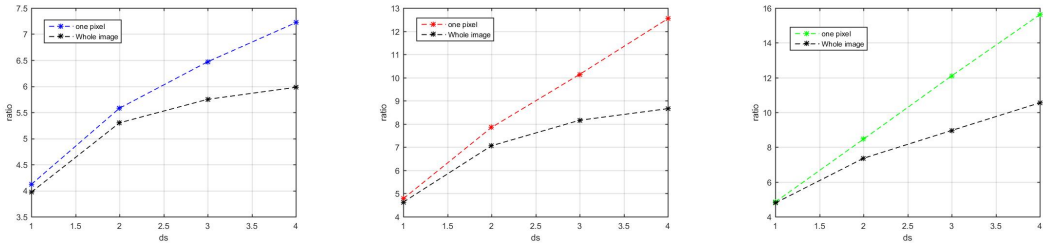


Figure 13: Ratio of RMSE, with different value of ds and Ds

We can see that in the Figure 13, with a larger size of patch window or research window, the RMSE will be reduced more after the filtering of non-local means. It is reasonable because in the background part with uniform intensity, a larger patch window will give higher possibility to obtain

a weight of 1 for each neighborhood, which is more similar to a mean filter. As for the research window, a larger size will increase the number of pixels in computing, which means the theoretical limit increases ($\sqrt{N}$ increases).

We also plot the curve of the ratio r for the whole image in function of ds, with $Ds = 10$, comparing with the ratio for one pixel:



(a) **(b)** **(c)**

Figure 14: Ratio of RMSE, for the whole image and for one pixel (a) Ds=5, (b) Ds=10, (c) Ds=15

In the Figure 12, we can observe the influence of spatial correlation between the pixels in the whole image, it will reduce the performance of filtering.

3.1.3 Noise reduction in OSC images

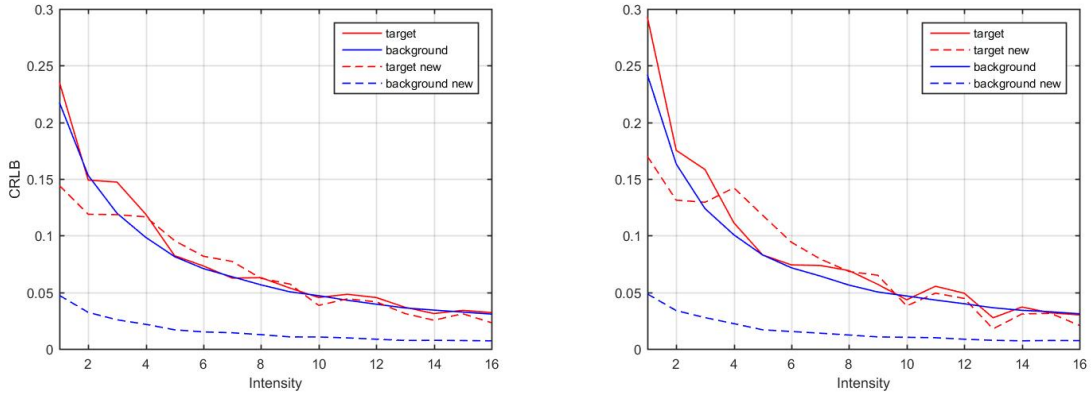
At the end, with the help of the mean square error we have obtained, we can calculate the CRLB of the OSC images with the formula:

$$CRLB = \frac{1}{N}\frac{(1+P^2)}{SNR^2} = \frac{1}{N}\frac{(1+P^2)}{(I/\sqrt{2}\sigma)^2}$$

where N=1 because we have only one measurement. In the Figure 15, we can see that the RMSE of X and Y are very close, so we define the total σ in the formula of $CRLB$ as:

$$\sigma = \frac{\sigma_X + \sigma_Y}{2}$$

so that we can obtain the curve of CRLB in function of different degree of intensities and we compare it with the variance which we compute from the value of background intensity:



(a) **(b)**

Figure 15: (a) CRLB, (b) RMSE of the target and the background computed from the noisy image(solid line) and the restored image(dashed line)

In the Figure 8, it is seen that for low intensities, the CRLB is a little smaller than the variance. As the intensity increase, the CRLB comes close to the variance in the DOP, either for the original DOP or the restored DOP.

We can also deduce that this non-local means denoising can reduce the RMSE of the background but it has little effect for the target because of its small size. That means in a low intensity, the target is difficult to find because its variance is still too large (variance of 0.15 while DOP is 0.5)

3.1.4 Optimization

In the non-local mean algorithm, the computing of weight depends on the Euclidean distance. When the signal to noise ratio is too low, the existence of noise will have a great impact on the computing. In order to decrease this impact, here we choose apply a pre-filtering before the non-local mean filtering, the sketch is shown below:

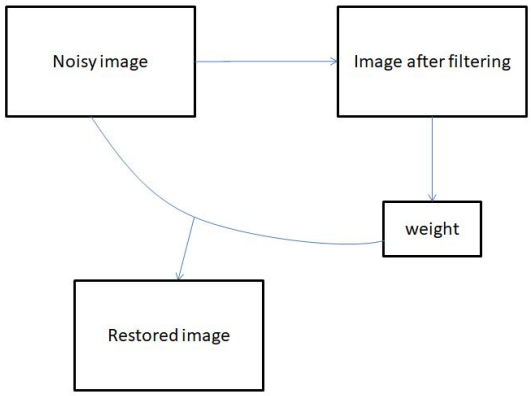
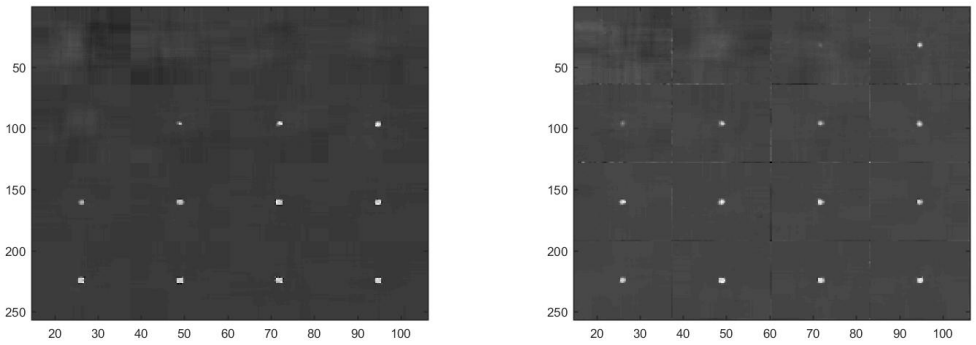


Figure 16: pre-filtering

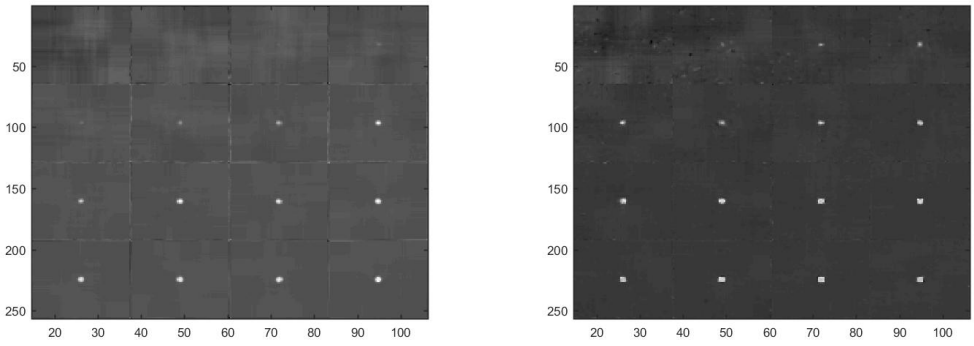
For pre-filtering, we denote the noisy image as Image A we apply a denoising filter to the noisy image and we can obtain a denoising image, this filter can be a classic filter such as mean filter or gaussian filter or this can be the non-local mean filter, we denote it as Image B. Then we calculate the weight for each pixel in Image A by using the value in Image B. And finally we can obtain a new restored image.

We apply this new method in the DOP figure before (squared images with different intensities and uniform background), the standard deviation of noise is also $\boxed{\sigma = 1}$. The results are shown in Figure 10 and Figure 11:



(a) **(b)**

Figure 17: Polarization image (a) pre-filtering, NL-means, (b) pre-filtering, mean filter with a size of 2×2



(a) **(b)**

Figure 18: Polarization image (a) pre-filtering, mean filter with a size of 3×3, (b) pre-filtering, gaussian filter with default parameters in Matlab

and we can also obtain their ratio of RMSE in image X and image Y and compare them with the ratio of RMSE with simple Non-local mean denoising :

Filter	NL-mean	NL-mean + NL-mean	Average 2×2 + NL-mean
ratio X	4.53	17.44	14.66
ratio Y	4.58	18.40	15.60
Filter	Average 3×3 + NL-mean	Average gaussian + NL-mean	
ratio X	12.28	13.34	
ratio Y	13.99	15.98	

we can see that the two methods can both improve the performance of denoising. The double NL-mean can achieve the highest ratio r, while the pre-filtering with a gaussian filter can provide a better contrast in low intensity. We can still distinguish the target in a very low intensity except the lowest one in Figure 11 (b)

3.2 Implementation in complex image

As we know, the Non-local means algorithm optimize the image with weighted value of neighborhood pixels, which is more adaptable to the complex condition.

Hence we choose a complex image to test the performance of our algorithm, which contains some bright parts as well as some dark parts. The original image is shown in the Figure **??** below:

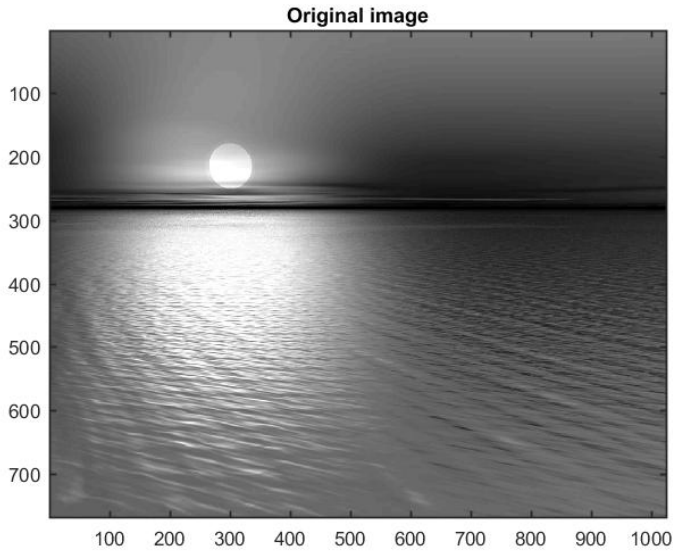


Figure 19: Original image

Computing the average of the pixel, we obtain a mean value of $\boxed{99.63}$, so we add a noise with its variance $\sigma = 10$, then the SNR of the whole image is:

$$SNR = \frac{I}{\sqrt{2}\sigma} = 7$$

The size of the image is 768×1024, In the simulation of GSC(general state contrast), we assume the average of the polarization is $P = 0.5$, we add a target with a size of 25×25 where the polarization difference is $\delta P = 0.2$. That means in the area of target, the polarization level is $P_t = 0.6$ and in the background we have $P_b = 0.4$, so we can define the intensity of polarization image:

$$I_X(i) = I(i) \cdot \frac{1+P}{2}$$

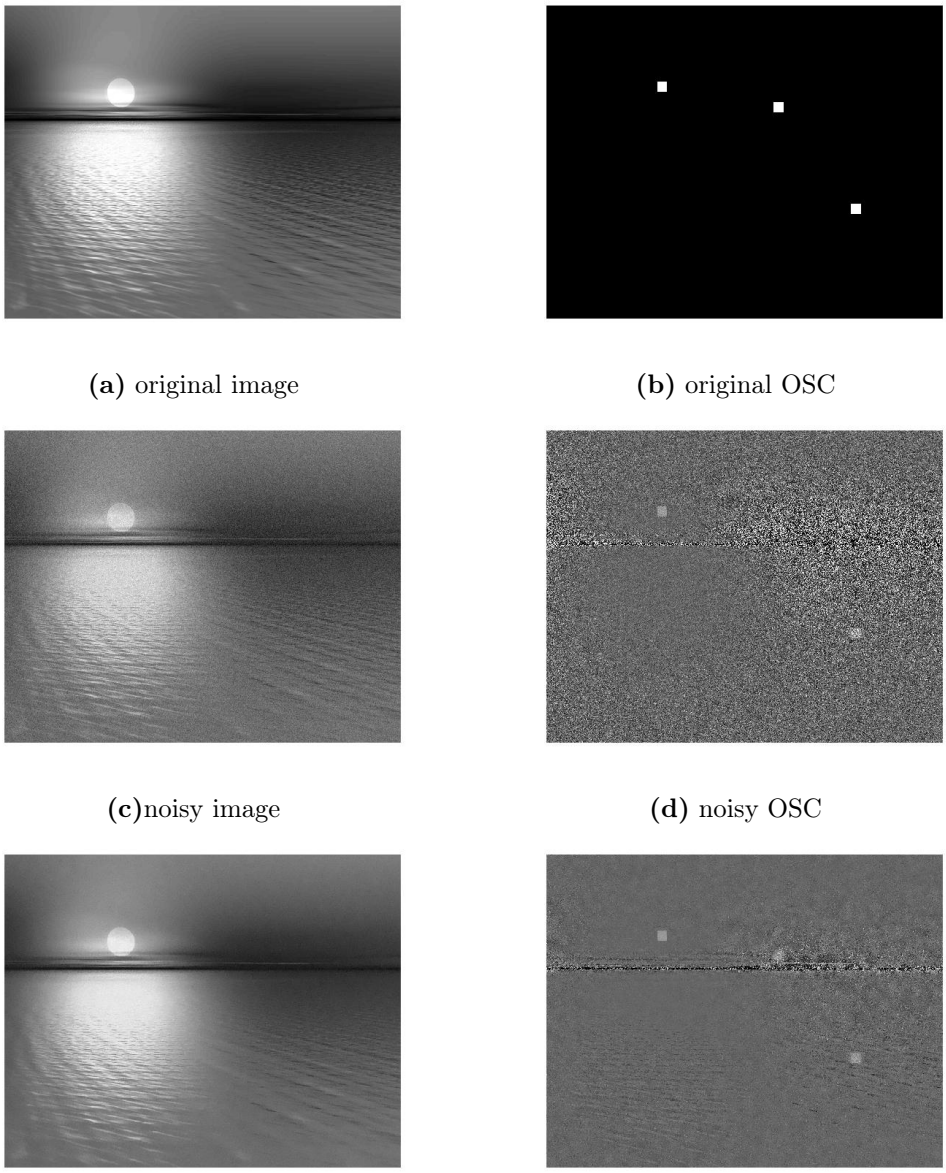
$$I_Y(i) = I(i) \cdot \frac{1-P}{2}$$

where P is P_t or P_b with respect to the target or background. And we can re-obtain the intensity of the image and the DOP :

$$I(i) = I_X(i) + I_Y(i)$$

$$P(i) = \frac{I_X(i) - I_Y(i)}{I_X(i) + I_Y(i)}$$

In our image, we place 3 targets in different intensity level, the result is shown in Figure **??**



(a) original image **(b)** original OSC

(c)noisy image **(d)** noisy OSC

(e) restored image **(f)** restored OSC

Figure 20: Implementation and comparison of NL-mean denoising in a complex image

Here, we introduce a threshold filter :

$$y(i) = \begin{cases} 0, & x(i) < 0 \\ x(i), & 0 \leq (i) \leq 1 \\ 1, & x(i) > 1 \end{cases}$$

in this case, the extreme values can be avoided in our DOP image. We can see that the distribution of noise in the OSC image is not gaussian. After non-local means denoising and threshold filtering, the background has been well denoised and the target is easier to distinguish. Meanwhile, the performance of filtering depends on the mean intensity around the target. A higher mean intensity will bring a better denoising effect. And with a low intensity, the non-local mean denoising can not distinguish well the target with the background noise.

In the next section, we will compare the result with the simple OSC image.

3.2.1 Comparison

As we know, the performance of the image X and image Y are similar, so in this section, we focus on the image X. We will compare the RMSE between the noisy image and restored image to study the performance of the non-local means algorithm in the OCS images. In the last section, we have seen that the noise in the OSC image is not evenly distributed, it depends on the partial intensity.

So in order to compare its performance, we choose a partial area which has twice size of the target(49×49). Then we rebuild a simple image which has the same size and same mean intensity as the partial area. Then we add the same target in the central of the simple image.

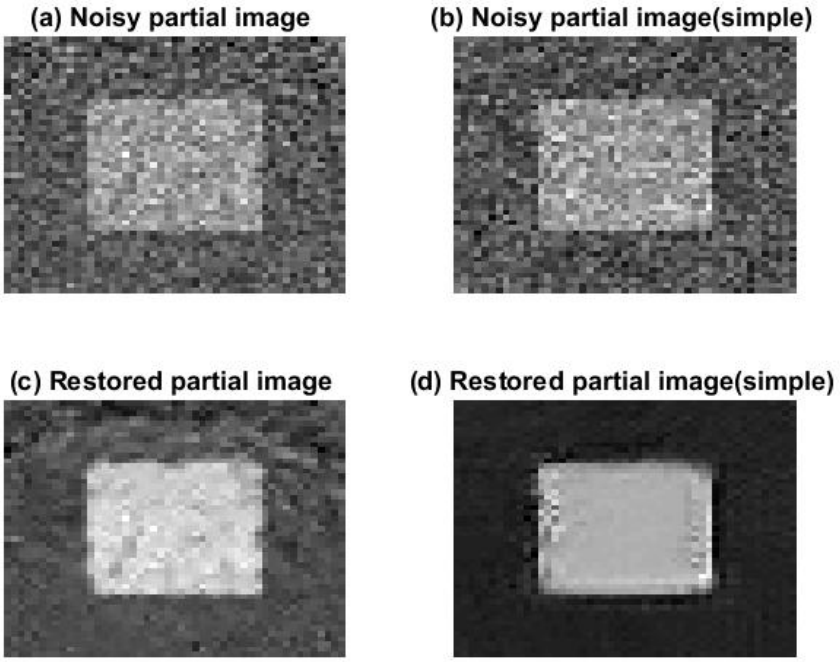


Figure 21: Comparison between complex image and simple image, with high intensity

to compare their performance, we define the RMSE:

$$RMSE = \sqrt{\frac{1}{N}\sum(u'(i) - u(i))^2}$$

where $u'(i)$ is the noisy pixel $\tilde{u}(i)$ or restored pixel $\hat{u}(i)$, $u(i)$ is the original pixel and N is the number of pixels in the image. And we also define the ratio of RMSE between the noisy image and restored image:

$$r = \frac{RMSE(\tilde{u})}{RMSE(\hat{u})}$$

where a higher ratio r means a better performance of the filtering in denoising.

With the help of *Matlab*, we can obtain the result of $RMSE$ and r and we put them in the table 3, we have also recognized that the mean value in the bright area is $\boxed{202}$, which means $\boxed{\text{SNR=14}}$

	Complex image			Simple image		
RMSE	noisy	restored	r	noisy	restored	r
X	10.10	3.92	2.58	9.94	2.01	4.94
P	0.0805	0.0281	2.86	0.0784	0.0207	3.78

Table 3: RMSE in bright area

we can see that the RMSE of restored complex image is higher than that in the simple image. That means the complexity in the image will reduce the performance of filter. Then, we repeat the operation in the area with middle intensity:

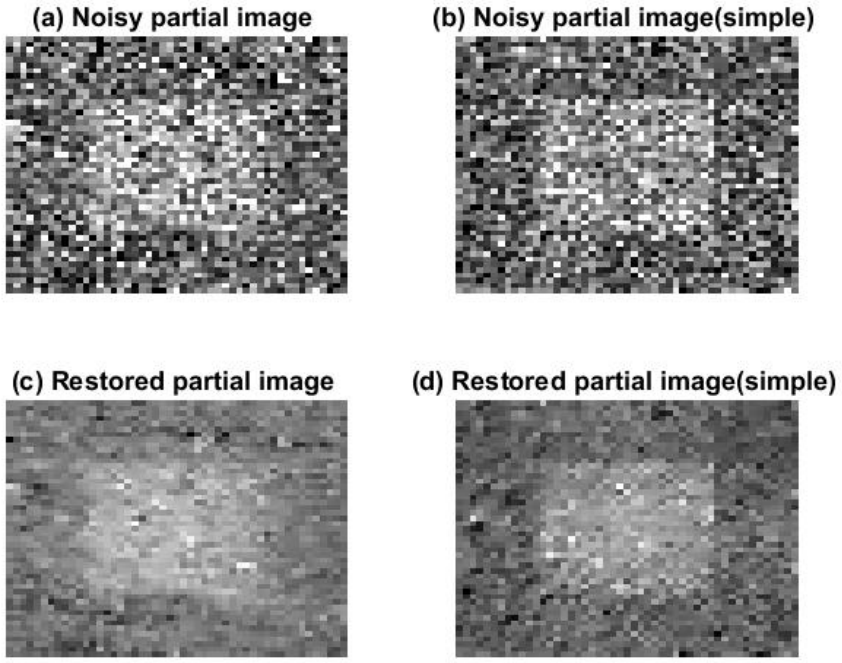


Figure 22: Comparison between complex image and simple image, with middle intensity

here, we have a mean intensity equals to $\boxed{71.5}$ and $\boxed{\text{SNR}=5.1}$, with a lower intensity, the target after filtering is more difficult to distinguish. But we can still find some contour of the target in the restored partial image. And the RMSE is shown below:

	Complex image			Simple image		
RMSE	noisy	restored	r	noisy	restored	r
X	9.92	5.33	1.86	9.89	2.80	3.53
P	0.228	0.0799	2.85	0.224	0.0678	3.30

Table 4: RMSE in middle area

Finally, we repeat the same operation in the dark area:

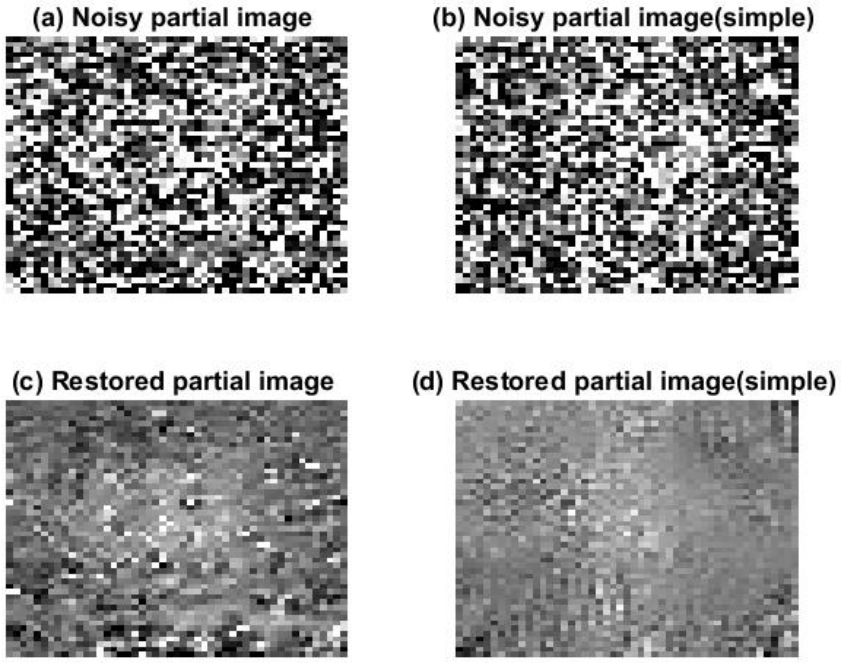


Figure 23: Comparison between complex image and simple image, with low intensity

in the dark area, the target is impossible to distinguish both in complex image and simple image. We recognize that in this case the mean intensity is $\boxed{27}$ and $\boxed{\text{SNR}=1.9}$, we can deduce that this algorithm is limited by SNR, the target can not be find in the restored image when the SNR is too low. We can also obtain its RMSE:

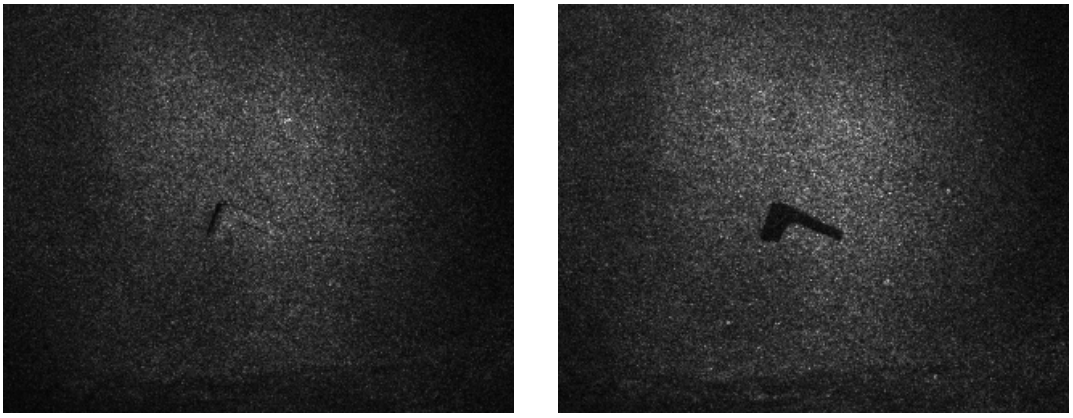
	Complex image			Simple image		
RMSE	noisy	restored	r	noisy	restored	r
X	9.73	2.89	3.36	9.99	1.28	7.80
P	0.386	0.166	2.32	0.382	0.0843	4.53

Table 5: RMSE in dark area

We can see that the RMSE in restored image is reduced with the decrease of SNR. Although the decrease ratio is even higher than that in bright area, however, the SNR is too low that the algorithm can not distinguish the target and background, which leads to such a result in Figure 23.

3.3 Implementation in practical images

Finally, we choose several practical OSC images obtained by polarimetric imaging setup and implement our denoising algorithm. As shown in the figures below, in the first image, we put a piece of metal with the shape of roundabout in the asphalt pavement:



(a) intensity image in horizontal direction **(b)** intensity image in perpendicular direction

Figure 24: The sequential intensity images

we can see this metal in the center of our images, which has the different polarization property from the environment. However, we can observe that there are much noise in our OSC images, so when we calculate its degree of polarization(DOP), we will obtain an image with a low contrast, as shown in Figure 25:

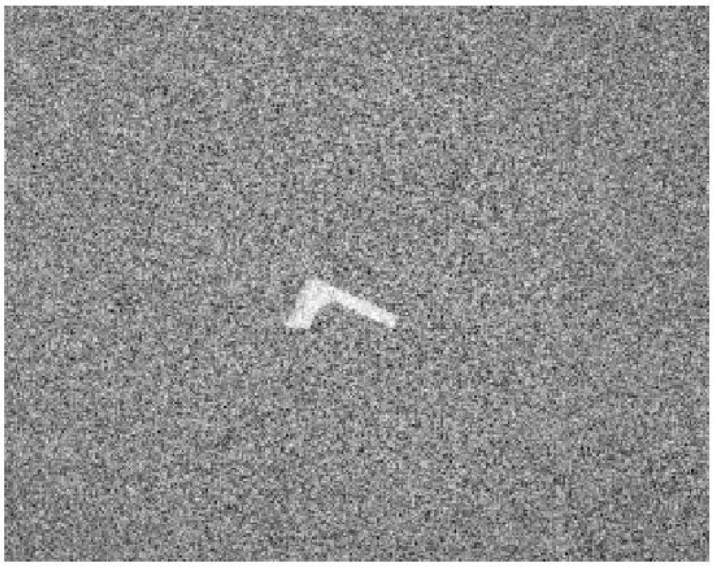
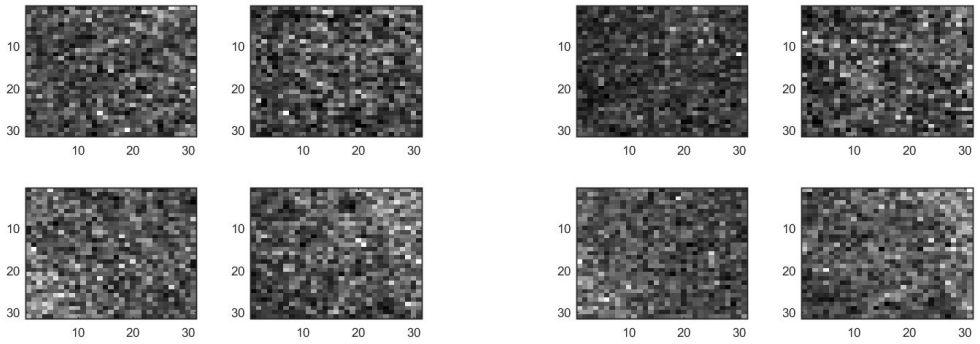


Figure 25: Noisy OSC image

3.3.1 Noise research

We have to recognize that in these intensity images, the noise may not be gaussian, so we should firstly study the type of noise. In the intensity images, we can find that there are some areas which have a quasi-uniform intensity. So we select some of these areas and then we calculate its average as the intensity and we calculate its variance as the level of noise in the area. In addition, the intensity values obtained by the camera are much larger than our maximum value(the maximum value of a pixel is 255, however the intensity values is mostly around 2000), so we divide all the intensity values by 100 to avoid over-saturation.

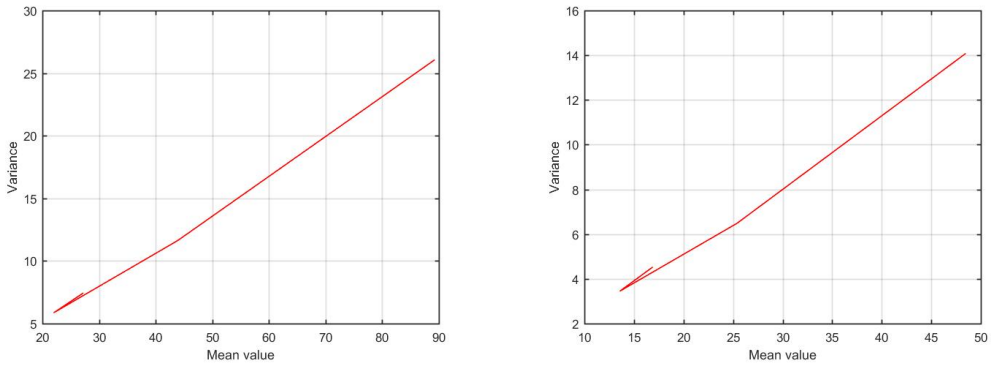
The following images show the quasi-uniform intensity areas we choose:



(a) intensity areas in horizontal direction **(b)** intensity areas in perpendicular direction

Figure 26: The quasi-uniform intensity areas

in each intensity image, we choose four areas with different average intensities, and we can plot the curve of variance of each area in function of average intensities:



(a) in horizontal intensity image **(b)** in perpendicular intensity image

Figure 27: Variance in function of average intensity

we can deduce that it is more like a Poisson noise. The level of noise will increase with the average intensity.

3.3.2 Implementation of NL-mean denoising

As we have discussed before, the parameters of NL-mean denoising depend on the level of noise, however the noise in our practical image has a distribution of Poisson, which means the intensity of noise changes with the average intensity of image. In order to fit for our algorithm, we choose the parameters with $\boxed{ds = 1}$, $\boxed{Ds = 10}$ and $\boxed{h = 4}$ in our following evaluation, which indicate a rough level of noise $\boxed{\sigma = 10}$.

In this part, we implement the non-local mean denoising algorithm with or without pre-filter. The first sample is the noisy OSC image we have used before, its intensity image is shown below:

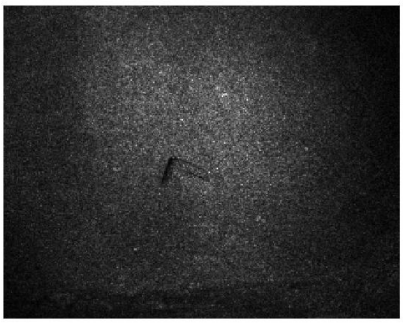
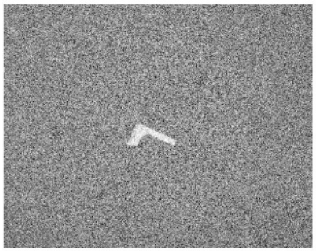


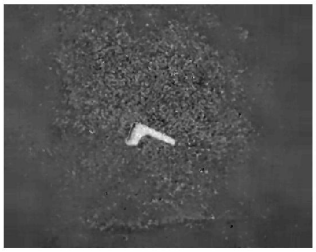
Figure 28: Intensity image

in order to optimize it, we implement 5 types of denoising filter: the NL-means denoising filter as well as NL-means denoising filter with different pre-filter, the results are shown in the Figure 29



(a) noisy OSC image

(b) restored OSC image using NL-mean



(c) restored OSC image, pre-filtering, average 2×2

(d) restored OSC image, pre-filtering, average 3×3

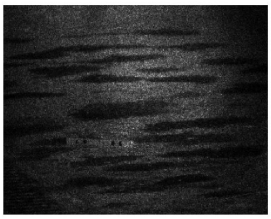
(e) restored OSC image, pre-filtering, gaussian

(f) restored OSC image, pre-filtering, double NL-mean

Figure 29: Implementation of NL-mean denoising, OSC image with a piece of metal in the asphalt pavement

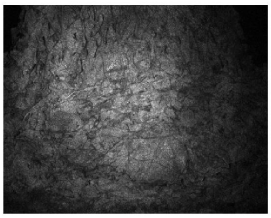
we can see that the NL-mean denoising filter can efficiently increase the contrast of OSC image. Simultaneously, the use of pre-filter can optimize the result of denoising. In the optimized OSC images, we can see that there exists the equalization in the four corners, this is due to the edge problem which has been discussed in section 1.5.

We also implement these denoising algorithm into other OSC images. Their intensity images are shown in Figure 30, and their OSC images, before or after filtering, are shown from Figure 31 to Figure 34.



(a) Metal A in asphalt road

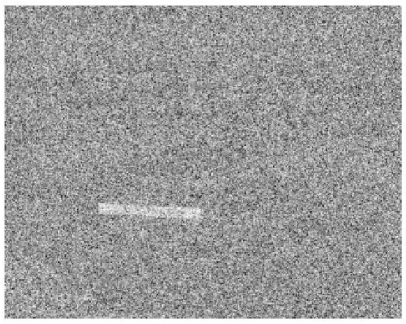
(a) Metal B in asphalt road



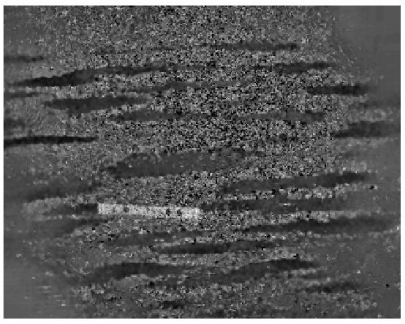
(c) Filet

(d) Metal behind plant

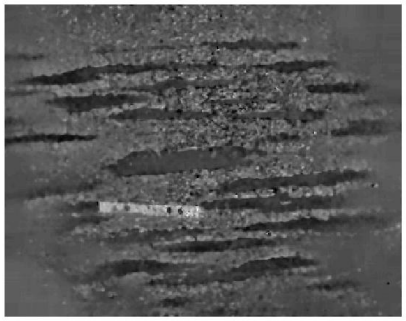
Figure 30: Intensity images



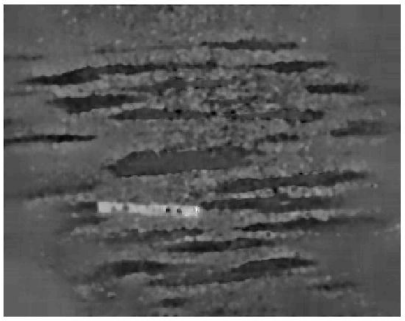
(a) noisy OSC image



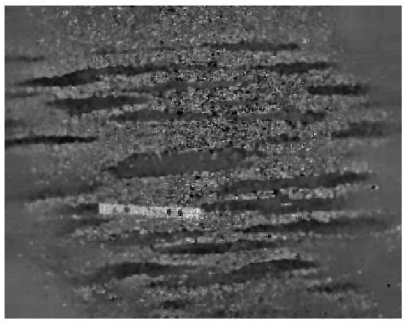
(b) restored OSC image using NL-mean



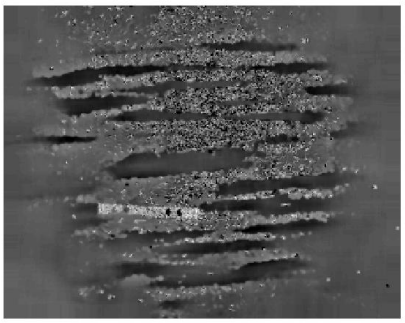
(c) restored OSC image, pre-filtering, average 2×2



(d) restored OSC image, pre-filtering, average 3×3

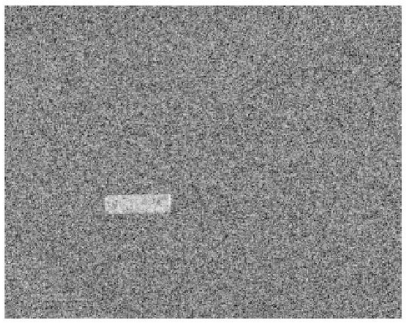


(e) restored OSC image, pre-filtering, gaussian

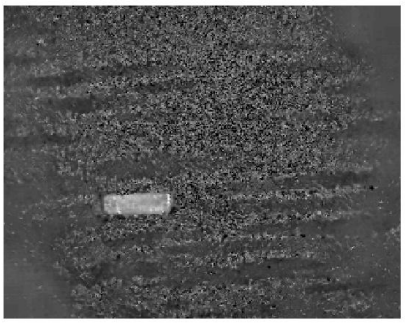


(f) restored OSC image, pre-filtering, double NL-mean

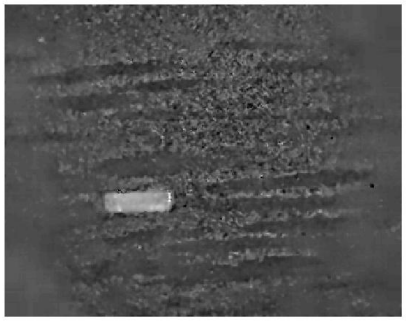
Figure 31: Implementation of NL-mean denoising, OSC image with metal A in the asphalt pavement with water



(a) noisy OSC image

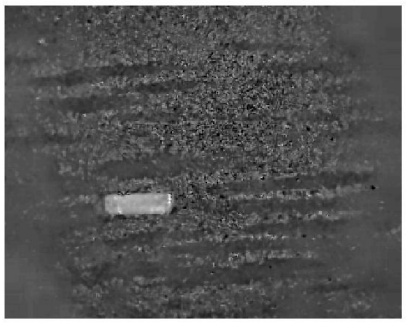


(b) restored OSC image using NL-mean

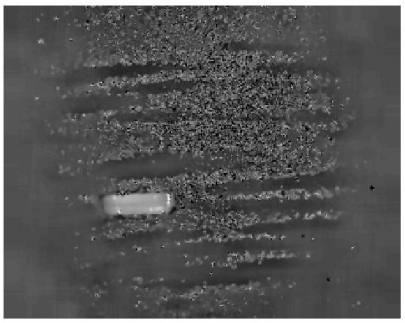


(c) restored OSC image, pre-filtering, average 2×2

(d) restored OSC image, pre-filtering, average 3×3

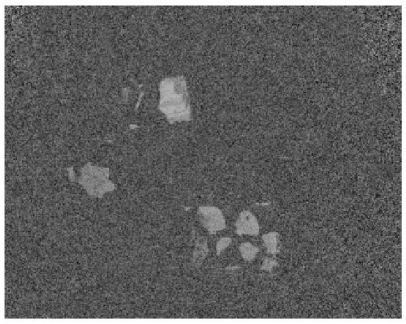


(e) restored OSC image, pre-filtering, gaussian

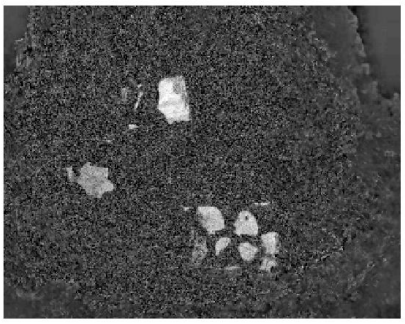


(f) restored OSC image, pre-filtering, double NL-mean

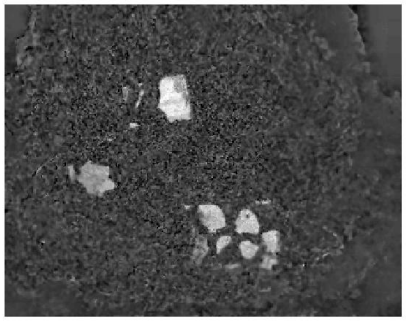
Figure 32: Implementation of NL-mean denoising, OSC image with metal B in the asphalt pavement with water



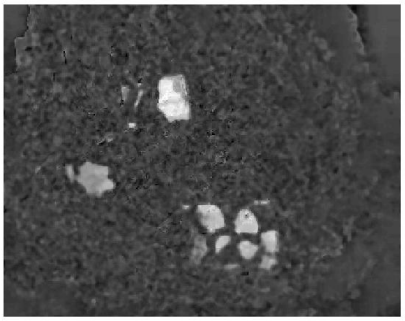
(a) noisy OSC image



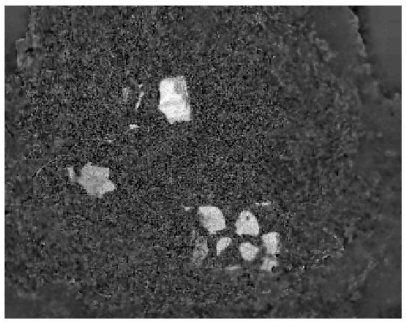
(b) restored OSC image using NL-mean



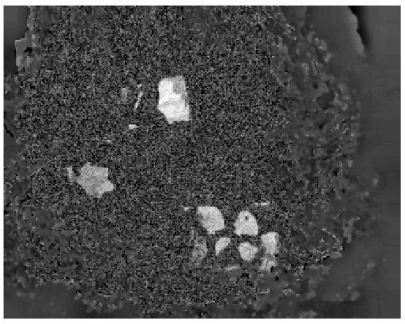
(c) restored OSC image, pre-filtering, average 2×2



(d) restored OSC image, pre-filtering, average 3×3



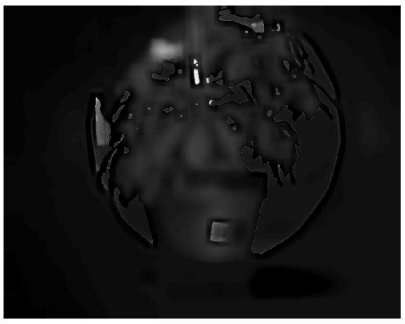
(e) restored OSC image, pre-filtering, gaussian



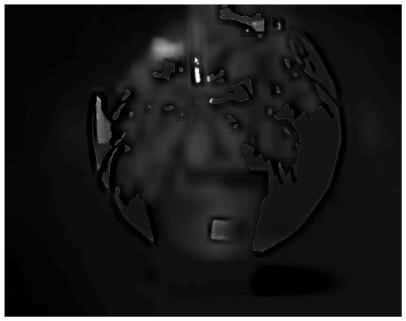
(f) restored OSC image, pre-filtering, double NL-mean

Figure 33: Implementation of NL-mean denoising, OSC image with metal B in the asphalt pavement with water

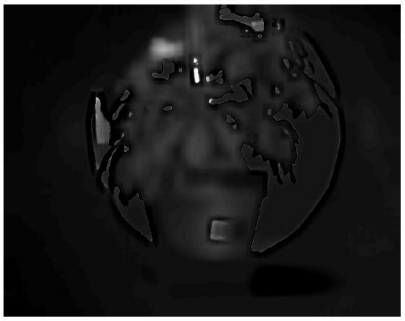
(a) noisy OSC image



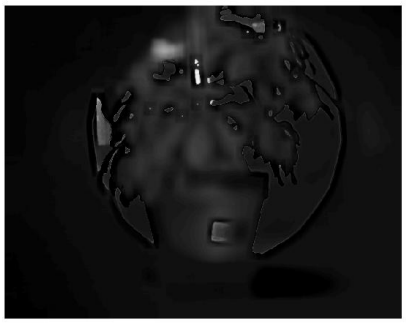
(b) restored OSC image using NL-mean



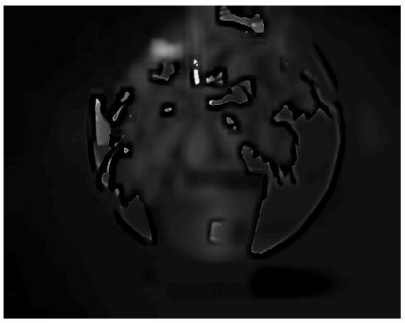
(c) restored OSC image, pre-filtering, average 2×2



(d) restored OSC image, pre-filtering, average 3×3



(e) restored OSC image, pre-filtering, gaussian



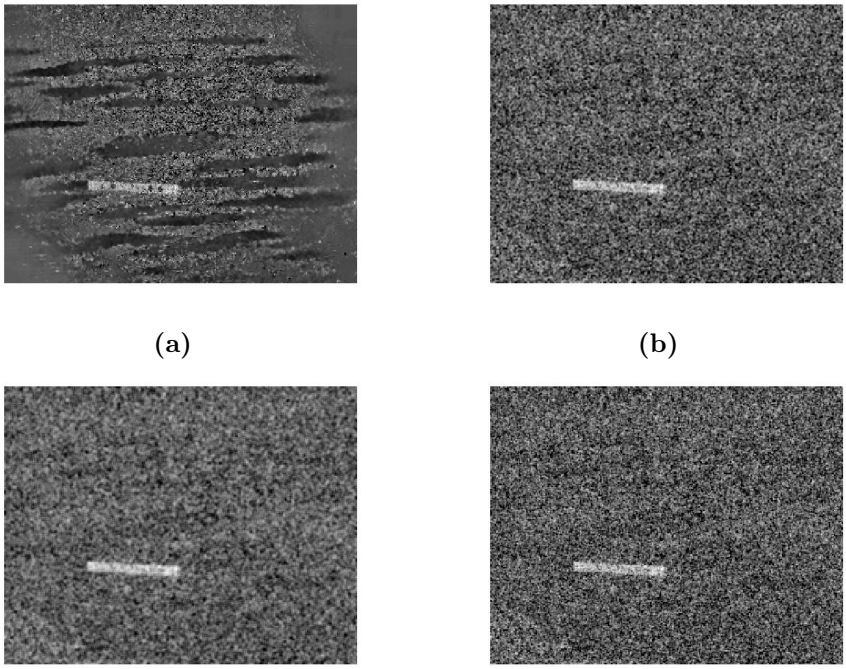
(f) restored OSC image, pre-filtering, double NL-mean

Figure 34: Implementation of NL-mean denoising, OSC image with metal B in the asphalt pavement with water

From these results, we can see that the NL-mean denoising filter can well allay the noise and increase the contrast between the target and background. However it is sensitive to the intensity of original image, it is reasonable as we have discussed in the section 3.1. When using the pre-filter, the edge of our OSC images are smoothed, and the noise of background has reduced, especially with a pre-filter of average. In some case, the double NL-mean denoising will enhance the contours of OSC images, we can observe it in Figure 34 (f).

Meanwhile, we find an extremely interesting phenomenon which exists in Figure 31 and Figure 32. We can see that in the noisy OSC images, we can roughly finger out a target and noisy background. However, in our optimized OSC images which use NL-mean denoising algorithm, we can find out not only the target, but also the water. We suppose that this is because the water also has the polarization property but it is too low to disguise in the noisy OSC images. With the NL-mean denoising algorithm, we can successfully reduce the noise and increase their contrast. So that we can see the water in restored OSC images.

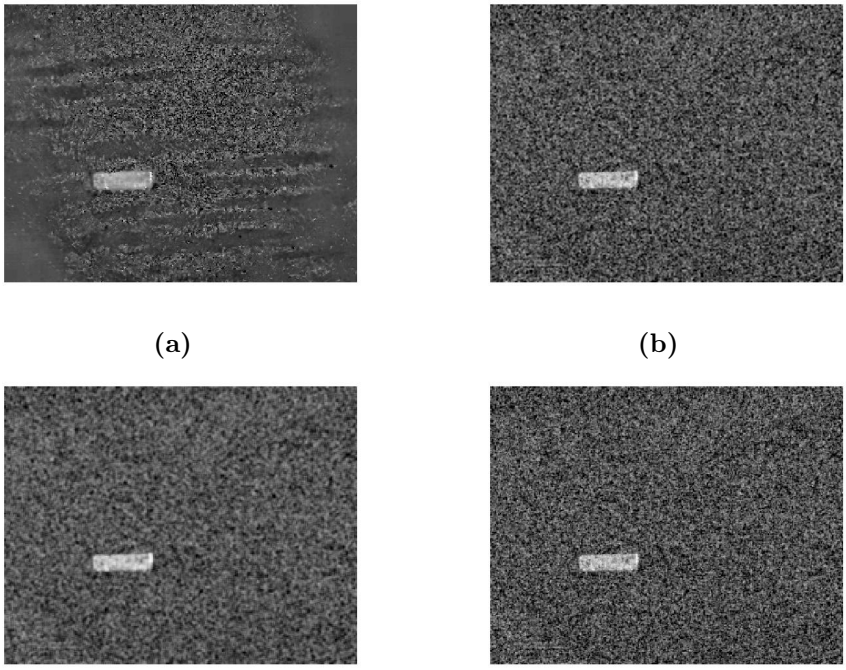
In order to verify our suppose, we use the traditional denoising filter to compare their results:



(a) **(b)**

(c) **(d)**

Figure 35: Denoising images, comparing with Figure 31 (a)NL-mean; (b)average,size 2×2; (c)average,size 3×3; (d)gaussian



(a) **(b)**

(c) **(d)**

Figure 36: Denoising images, comparing with Figure 32 (a)NL-mean; (b)average,size 2×2; (c)average,size 3×3; (d)gaussian

We could also find some traces of water in these images. Hence, we can confirm that the water does have the polarization property and the NL-mean can well denoising it.

In the meantime, we find that in the Figure 34, some areas have been blurred after filtering. We have to recognize that the noise in the practical OSC images is not gaussian, which means its variance is not constant. However, in the previous discussion, we define the parameters of NL-means denoising algorithm according to the variance of noise. And in the denoising of the Plant image, we assume the variance is 10. We suppose that the decrease of contrast is due to the selection of variance. In the figures below, we choose a small area which contains a polaziser and we change the assumed variance:

(a) I_{par} **(b)** I_{per} **(c)** I_{OSC}

Figure 37: Original images

The standard derivation of polarizer is 0.013.

(a) I_{par} **(b)** I_{per} **(c)** I_{OSC}

Figure 38: ds=1, Ds=10, h=0.4, $\sigma_{assumed} = 1$

The standard derivation of polarizer is 0.008.

(a) I_{par} **(b)** I_{per} **(c)** I_{OSC}

Figure 39: ds=1, Ds=10, h=0.8, $\sigma_{assumed} = 2$

The standard derivation of polarizer is 0.012.

(a) I_{par} **(b)** I_{per} **(c)** I_{OSC}

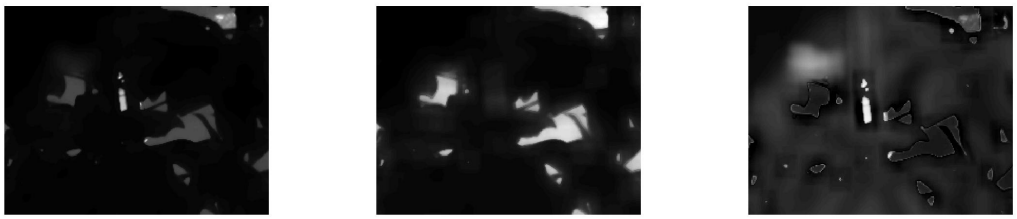
Figure 40: ds=1, Ds=10, h=1.2, $\sigma_{assumed} = 3$

The standard derivation of polarizer is 0.017.

(a) I_{par} **(b)** I_{per} **(c)** I_{OSC}

Figure 41: ds=1, Ds=10, h=2, $\sigma_{assumed} = 5$

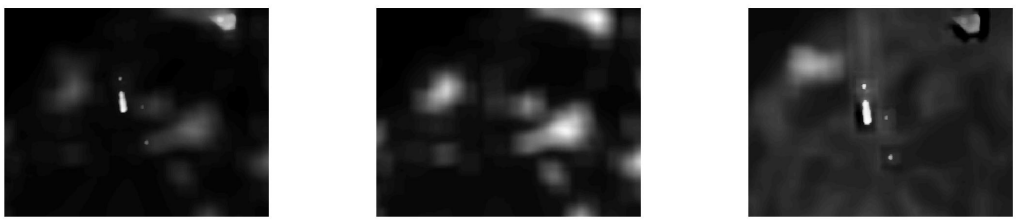
The standard derivation of polarizer is 0.0299.



(a) I_{par} **(b)** I_{per} **(c)** I_{OSC}

Figure 42: ds=1, Ds=10, h=4, $\sigma_{assumed} = 10$

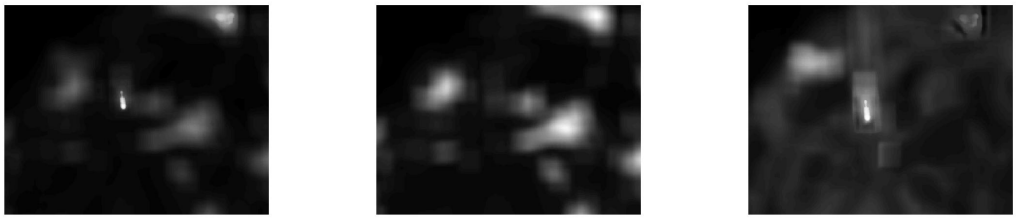
The standard derivation of polarizer is 0.0353.



(a) I_{par} **(b)** I_{per} **(c)** I_{OSC}

Figure 43: ds=2, Ds=10, h=8, $\sigma_{assumed} = 20$

The standard derivation of polarizer is 0.0353.



(a) I_{par} **(b)** I_{per} **(c)** I_{OSC}

Figure 44: ds=2, Ds=10, h=12, $\sigma_{assumed} = 30$

We can see that when we choose a small assumed σ, the phenomenon of blur has decreased, which confirms our assumption.

References

[1] F. Goudail and Ph. Réfrégier. Statistical techniques for target detection in polarization diversity images. *Opt. Lett.*, 26(9):644–646, May 2001.

[2] Arnaud Bénière, François Goudail, Mehdi Alouini, and Daniel Dolfi. Precision of degree of polarization estimation in the presence of additive gaussian detector noise. *Optics Communications*, 278(2):264–269, 2007.

[3] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 60–65. IEEE, 2005.

[4] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. Non-local means denoising. *Image Processing On Line*, 1:208–212, 2011.

[5] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. Nl-means denoising. http://demo.ipol.im/demo/bcm_non_local_means_denoising/, 2011.