Telecom-Paris

Travaux Pratiques - IMA201

TP2

Filtrage et Restauration

Réalisé par :
Baccari Mohamed Mootez

2.Transformation géométrique

- On va appliquer une rotation de 40° sur l'image « lena.tif » par les deux méthodes : PPV et méthode bilinéaire

Différence entre les deux images :

La transformation géométrique n'est autre qu'une interpolation faite sur une image.
Pour une valeur non entière :

- La méthode de Plus proche voisin (PPV) attribue la valeur du pixel du plus proche voisin pour faire le remplissage de ce point.
- La méthode de l'interpolation bilinéaire, elle prend en considération les valeurs des 4 voisins les plus proches de cette valeur pour faire l'affectation de la valeur donnée.

- L'interpolation bilinéaire permet d'avoir une image moins pixelisé que celle obtenue à partir de la méthode de PPV.
  - On va appliquer huit rotations de 45° sur l'image « lena.tif » par les deux méthodes : PPV et méthode bilinéaire.

La transformation géométrique n'est autre qu'une interpolation faite sur une image.
En effet, les cordonnées trouvés ne sont pas nécessairement des entiers

- Importance de l'interpolation pour effectuer cette transformation géométrique correctement.

On remarque que la rotation par le PPV donne une image très pixelisée. Dans le PPV, on prend en considération que le plus proche voisin. Par contre pour la transformation bilinéaire l'image est beaucoup plus nette, ceci revient au fait qu'elle prend en considération les 4 plus proche voisins pour faire l'affectation de la valeur manquante à ce pixel.

- L'effet de pixélisation par le PPV est beaucoup plus visible que lorsqu'on a appliqué une seule rotation. En effet, chaque rotation est une interpolation et donc on va à chaque fois considérer que le PPV, ce qui dégrade l'image.

  - Application d'une rotation de 360° à Lena après avoir appliqué un facteur de zoom inférieur à 1

On remarque l'apparition du phénomène d'aliasing sur l'image de « lena ». Il y'a eu un sous échantillonnage générant un repliement de spectre qui est très visible au niveau des cheveux de « lena ».
Afin d'atténuer ce phénomène on applique un filtre passe bas avant de faire le sous l'échantillonnage.

3.Filtrage linéaire et médian

- D'après la définition du noyau du filtre gaussien :

$$ss = int(max(3, 2 * np.round(2.5 * s) + 1))$$

C'est-à-dire, si on a s <=3/5, le noyau du filtre est égal à 3 fois la déviation standard (s) qui détermine l'ouverture de la cloche gaussienne. Sinon, la taille du noyau de filtre gaussien prend la valeur de $2 * np.round(2.5 * s) + 1$ qu'est relativement grande.

- La variance de l'image initiale sans bruit est égale à 0.

Généralement, une image contient plusieurs zones qui ne sont pas homogènes, pour cette raison, on ne peut pas calculer la variance sur toute une image pour comparer la quantité de bruit résiduel avant et après application des filtres linéaires à une image bruitée.

Pour cette raison, on va sélectionner une zone uniforme dans une image et calculer la variance dans cette zone pour les 3 images : bruitée, filtrée avec un filtre gaussien, filtrée avec un filtre à noyau constant.

On remarque qu'après l'opération de filtrage, le bruit devient très réduit. La variance de l'image passe de 99 à 0.17 pour le filtre de gauss et 3,25 pour le filtre à noyau constant.

On remarque qu'à l'aide du noyau du filtre gaussien, on élimine mieux le bruit.

- Comparaison linéaire/médian sur l'image « pyra-impulse.tif »

Le filtre linéaire fait la somme de toutes les valeurs qui se trouvent dans un patch donné puis renvoi une valeur qui combine toutes ces valeurs. Par contre, pour un filtre médian, il fait le tri entre toutes les valeurs qui se trouve dans le patch et sélectionne la valeur qui se trouve exactement au milieu. De ce fait, le filtre médian ignore totalement les valeurs extrêmes et donc élimine le bruit.

Donc Pour une image qui ne présente pas trop de bruit (comme dans notre cas), le filtre médian est un filtre très efficace pour l'élimination des petites taches qui se présentent sur une image. Ceci est très visible sur l'image « pyra-impulse.tif ». On voit clairement que le filtre médian donne une image plus nette que les autres filtres linéaires. Le filtre gaussien élimine totalement le bruit mais il renvoie une image avec beaucoup de flou. Quant au filtre à noyau constant, le bruit est encore présent dans l'image.

- Différence de comportement sur le point lumineux de « carre_orig.tif »

Le filtre médian donc élimine totalement le point lumineux qui se situe en haut à droite de l'image « carre_orig.tif ». En effet, le filtre médian sélectionne la valeur qui se trouve au milieu, il élimine donc les valeurs extrêmes.

Le point lumineux correspond à une valeur extrême dans cette zone, c'est pour cela que le filtre médian a totalement éliminé ce point. On remarque aussi que l'extrémité du carré devient plus lisse avec une courbure plus importante.

Par contre pour le filtre linéaire constant, on voit clairement un petit point moins lumineux qui correspond à la moyenne des pixels dans cette zones.

4.Restauration :

- Filtrage linéaire et filtrage inverse d'une image avec ou sans ajout de bruit

  On remarque que lorsqu'on filtre l'image de Lena par un filtrage linéaire et on applique le filtre inverse pour restaurer l'image, on obtient exactement la même image de départ. Par contre, si on ajoute un petit bruit à l'image filtré, ce qui est le cas dans la vie réelle, (bruit de quantification ou de mesure..) , on perd totalement l'information .

- Détermination du noyau de convolution qu'a subi l'image

Le noyau de convolution est appliqué à toute l'image, or dans l'image carré, on remarque qu'il y a un petit point lumineux (une sorte d'impulsion) qui se trouve en haut à gauche. Donc, déterminer le noyau appliqué à l'image revient à voir la transformation subit par cette impulsion.

On peut facilement remarquer que le noyau du filtre appliqué à l'image est :

0	0	1/5
1/5	1/5	1/5
0	0	1/5

- On s'intéresse maintenant au filtrage de Wiener. On connait déjà le noyau du filtre. On va ajouter donc du bruit à l'image avant de la passer au filtre de Wiener.

On remarque que lorsque λ est faible (pour 0 et 0.1) on n'a pas une bonne restauration. Lorsqu'on augmente λ l'image originale commence peu à peu à être rétablit. On remarque qu'il y a le phénomène de sous-estimation sous un certain seuil et le phénomène de sur estimation lorsque λ dépasse 14, l'image devient plus floue.

⇨ La meilleure image restaurée qu'on a obtenue est celle-ci lorsqu'on a pris une valeur de lambda proportionnelle au bruit.

5.Applications

- Comparaison filtrage linéaire et médian

On applique le filtre médian à l'image. L'image obtenu n'est autre que le produit de convolution entre l'image initiale et le filtre médian. On sait que le filtre médian n'a pas de Kernel, il prend les valeurs qui se trouve dans un patch donné, il les regroupe et prend la valeur qui se trouve au milieu, donc le kernel qu'on va obtenir sera un kernel d'un filtre linéaire équivalent au filtre médian.

Mais cette approche ne va pas nous garantir que le kernel du filtre qu'on va obtenir est celui du filtre linéaire a noyau constant. Pour cette raison, nous allons faire des itérations sur le filtre à noyau constant en changent à chaque fois la taille de ce noyau et nous allons considérer la variance de l'image totale. On va donc prendre la taille du noyau du filtre constant la plus bas qui minimise la différence de variance entre lui et celle du filtre médian.

```
im=skio.imread('images/carre_orig.tif')
im_noise=noise(im,5)
im_median=median_filter(im_noise,r=4)
var=var_image(im_median,0,0,256,256)
mini=1e9
condidate=0
for i in range(1,11,2):
    ker_gauss_cst=get_cst_ker(i)
    im_noise=noise(im,5)
    var_cst=var_image(filtre_lineaire(im_noise,ker_gauss_cst),0,0,256,256)
    if(np.abs(var_cst-var)<mini):
        mini=np.abs(var_cst-var)
        condidate=i
print('Le filtre lineaire de noyau constant qui sera l'' equivalent dans notre cas est de taille',condidate)
```

On obtient l'Output suivant :

```
Le filtre lineaire de noyau constant qui sera l equivalent du filtre médian est de taille 3
```

- Calcul théorique du paramètre de restauration

Le changement consiste à modifier seulement le dénominateur de la fonction du filtre de Wiener comme indiqué dans l'énoncé.

```
image= skio.imread('images/carre_flou.tif')
imagebr = noise(im,10)
variancebr = 100 *256 *256
variance = np.fft.fft2(imbr) ** 2
K = np.array([[0,0,1/5],[1/5,1/5,1/5],[0,0,1/5]])
im_neww = wiener_new(imbr,K,var,varb)
```