y
DIRO
IFT 6150

EXAMEN THÉORIQUE

Max Mignotte

DIRO, Département d'Informatique et de Recherche Opérationnelle
http: //www.iro.umontreal.ca/∼mignotte/ift6150
e-mail: mignotte@iro.umontreal.ca

Date: *15/12/2025*

I	Transformée de Fourier (28 pts)
II	Spectre & Filtrage (23 pts)
III	Restoration (16 pts)
IV	Codage en C (33 pts)
V	Misc. & Questions relatives aux TPs (17 pts)
Total	117 points

Tous documents, calculatrices et calculateurs personnels autorisés

I. Transformée de Fourier (28 pts)

Soit $f(x)$, une fonction (ou un signal spatial) et $F(\nu)$ sa Transformée de Fourier (TF) ou son spectre, *i.e.*; $f(x) \rightleftharpoons F(\nu)$ où $\mathcal{F}[f(x)] = F(\nu)$. Dans les questions de cet exercice, nous allons essayer de trouver la TF de la fonction $\frac{1}{x}$ de différentes façons.

1. Énumérer **toutes** les propriétés de $F(\nu)$, la TF de la fonction $f(x) = \frac{1}{x}$ (sans la calculer).
   <4 pts>

2. Dans les tables de Transformée de Fourier (TF), on trouve facilement: $\text{sgn}(x) \overset{\mathcal{F}}{\rightleftharpoons} \frac{1}{j\,\pi\,\nu}$
   où $\text{sgn}(x)$ est la fonction signe ($=1\ \forall\, x>0,\ \ =-1\ \forall\, x<0$, et $=0\ \ \forall\, x=0$).
   En utilisant la propriété duale de cette TF, trouver la TF de $f(x) = \frac{1}{x}$.
   <4 pts>

3. Dans les tables de Transformée de Fourier (TF), on trouve aussi: $\mathcal{H}(x) \overset{\mathcal{F}}{\rightleftharpoons} \frac{1}{2}\left[\delta(\nu) + \frac{1}{j\,\pi\nu}\right]$
   où $\mathcal{H}(x)$ est la fonction d'Heaviside (définie par $H(x)=1$ si $x>0$ et 0 ailleurs).
   En utilisant la propriété duale de cette TF, trouver la TF de $f(x) = \frac{1}{x}$.
   <4 pts>

4. Dans les tables de Transformée de Fourier (TF), on trouve aussi: $\ln|x| \overset{\mathcal{F}}{\rightleftharpoons} \frac{-\,\text{sgn}(\nu)}{2\,\nu}$
   En utilisant une propriété sur cette TF (autre que la propriété duale), trouver la TF de $f(x) = \frac{1}{x}$.
   <4 pts>

5. Dans les tables de Transformée de Fourier (TF), on trouve aussi: $\frac{1}{a^2+x^2} \overset{\mathcal{F}}{\rightleftharpoons} \frac{\pi}{a}\exp(-2\pi\,a\,|\nu|)$
   En utilisant la TF de $x\,f(x)$ et la limite de ce produit quand $f(x)=\frac{1}{a^2+x^2}$ quand a tend vers quelque chose que l'on définira, trouver la TF de $f(x) = \frac{1}{x}$.
   <4 pts>

6. Dans les tables de Transformée de Fourier (TF), on trouve aussi: $\frac{\exp(-a\,|x|)}{x} \overset{\mathcal{F}}{\rightleftharpoons} -2\,j\cdot\arctan\left(\frac{2\,\pi\,\nu}{a}\right)$
   Utiliser cela pour trouver la TF de $f(x) = \frac{1}{x}$.
   <4 pts>

7. Si un signal $s(x)$ était convolué avec le signal $f(x) = (1/x)$, avec quelle sorte de filtre, le signal $s(x)$ serait-il filtré ?
   <4 pts>

Réponse

1.

La fonction $f(x) = (1/x)$ est non périodique donc son **spectre sera continu**. Elle est réelle et impaire donc son spectre sera **imaginaire pur et impair** (à cause de la symétrie hermitienne), avec donc des fréquences négatives. Notons aussi qu'en $x = 0$, sa limite est $\pm\infty$ et est différente à gauche et à droite créant ainsi une discontinuité très grande en ce point. $F(\nu)$ aura donc un spectre **avec un très grand support fréquentiel**, c'est-à-dire qu'il faudra beaucoup d'harmoniques pour reconstruire cette fonction. De plus, cette fonction n'est pas intégrable (singularité en 0) et, de ce fait, le spectre sera **discontinu à l'origine** (*i.e.*, en $\nu = 0$) et on ne peut utiliser la définition de la TF sous sa forme intégrale (donc non calculable par TF analytique) ce qui nous renseigne que celle-ci sera forcement **constituée d'une ou plusieurs distributions**.
<4 pts>

2.

En utilisant la propriété duale sur $\mathcal{F}[\text{sgn}(x)] = \frac{1}{j\,\pi\,\nu}$, on trouve: $\mathcal{F}^{-1}[\text{sgn}(\nu)] = \frac{1}{-j\,\pi\,x}$ et donc:

$$\mathcal{F}\left[\frac{1}{x}\right] = -j\,\pi\,\text{sgn}(\nu)$$

<4 pts>

NOTA : On vérifie bien que le spectre $F(\nu) = -j\,\pi\,\text{sgn}(\nu)$ obtenu est bien continu, imaginaire pur et impair, avec un support fréquentiel très grand et une discontinuité à l'origine (cf. Q1.(a)).

3.

En utilisant la propriété duale sur $\mathcal{F}[\mathcal{H}(x)] = \frac{1}{2}\left[\delta(\nu) + \frac{1}{j\,\pi\nu}\right]$, on trouve ▷ $\mathcal{F}^{-1}[\mathcal{H}(\nu)] = \frac{1}{2}\left[\delta(x) - \frac{1}{j\,\pi x}\right]$ (formule que l'on peut directement prendre aussi des anciens Examens finaux IFT6150).

Puis en calculant la TF à gauche et à droite de cette expression, on a directement:

$$\begin{aligned} \mathcal{H}(\nu) &= \frac{1}{2} - \frac{1}{2\,j\,\pi}\,\mathcal{F}\left[\frac{1}{x}\right] \\ \mathcal{F}\left[\frac{1}{x}\right] &= -j\,\pi\,\underbrace{(2\,\mathcal{H}(\nu) - 1)}_{\text{sgn}(\nu)} = -j\,\pi\,\text{sgn}(\nu) \end{aligned}$$

<4 pts>

4.

En utilisant la propriété: $\mathcal{F}[f'(x)] = 2\pi j\nu \cdot F(\nu)$ et le fait que: $\big(\ln|x|\big)' = \frac{1}{x}$, on trouve facilement:

$$\mathcal{F}\big[\big(\ln|x|\big)'\big] = \mathcal{F}\left[\frac{1}{x}\right] = 2\pi j\nu \cdot \mathcal{F}\big[\ln|x|\big] = 2\pi j\nu \cdot \left(-\frac{\text{sgn}(\nu)}{2\,\nu}\right) = -\pi j\,\text{sgn}(\nu)$$

<4 pts> donc: $\mathcal{F}\left[\frac{1}{x}\right] = -\pi j\,\text{sgn}(\nu)$

5.

En utilisant la propriété: $\mathcal{F}[f'(x)] = 2\pi j\nu \cdot F(\nu)$ et la propriété duale, on trouve $\mathcal{F}[xf(x)] = \frac{F'(\nu)}{-2\,\pi\,j}$

En utilisant cette propriété pour calculer la TF de $x \cdot \frac{1}{a^2+x^2}$ on trouve:

$$\begin{aligned} \mathcal{F}\left[x \cdot \frac{1}{a^2+x^2}\right] &= -\frac{1}{2\pi\,j} \cdot \left(\frac{\pi}{a}\Big[\exp\left(-2\,\pi\,a\,|\nu|\right)\Big]\right)' \\ &= -j\,\pi \cdot \exp\left(-2\,\pi\,a\,|\nu|\,\right) \cdot \text{sgn}(\nu) \end{aligned}$$

<4 pts> Et lorsque $a \to 0$, on peut écrire: $\mathcal{F}\left[\frac{1}{x}\right] = -\pi j\,\text{sgn}(\nu)$

6.

En faisant tendre $a \to 0$, on obtient (sachant que: $\arctan(\pm\infty) = \pm\pi/2$):

$$\mathcal{F}\left[\frac{1}{x}\right] = \begin{cases} -j\,\pi & \text{si } \nu > 0 \\ j\,\pi & \text{si } \nu < 0 \end{cases} = -\pi\,j\,\text{sgn}(\nu)$$

<4 pts>

7.

En première approximation, on pourrait dire qu'il s'agit d'un filtre de la famille des filtres Passe-bas (puisque tous les coefficients discrets de $1/x$ pour $x > 0$ sont positifs) (notons que dans le cas discret, on devra donner une valeur en 0, certes grande, mais non infinie pour que ce filtre numérique soit numériquement applicable et implémentable).

Mais plus précisément, calculons le module (cf. Question 6. et du fait que $|j| = 1$):

$$\left|\mathcal{F}\left[\frac{1}{x}\right]\right| = \pi$$

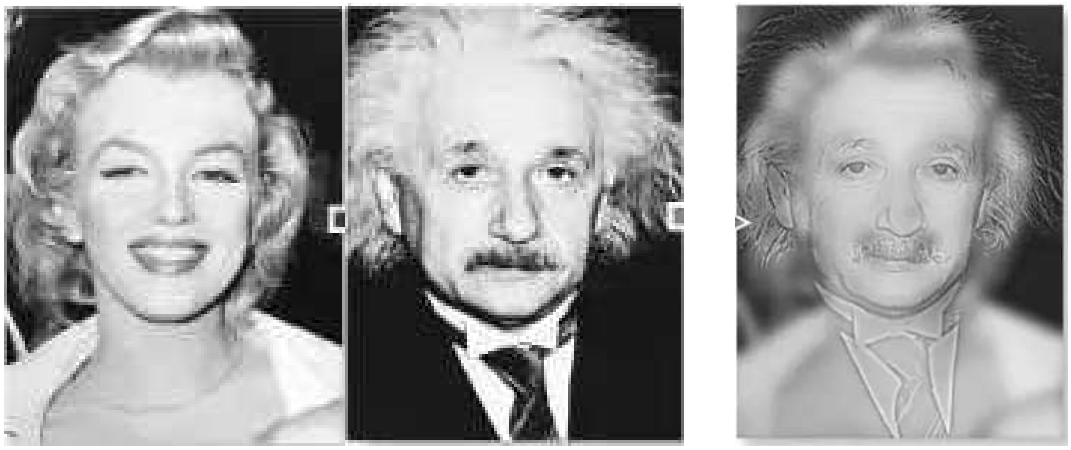
Le module est constant et ne modifiera pas (différemment selon les fréquences) les amplitudes des fréquences du signal filtré (ni atténuation, ni amplification). Bref le filtre est certes de la famille des filtres de basse

fréquence (en première approximation) dans le sens qu'il laisse passer les basses fréquences et atténue (ici pas du tout) les hautes fréquences. Seule la phase du signal filtré sera modifiée. Le signal n'est ni lissé, ni amplifié, il est seulement déphasé (de $(-\pi/2)$ pour $\nu > 0$ et de $(\pi/2)$ pour $\nu < 0$).
<4 pts>

II. Spectre & Filtrage (23 pts)

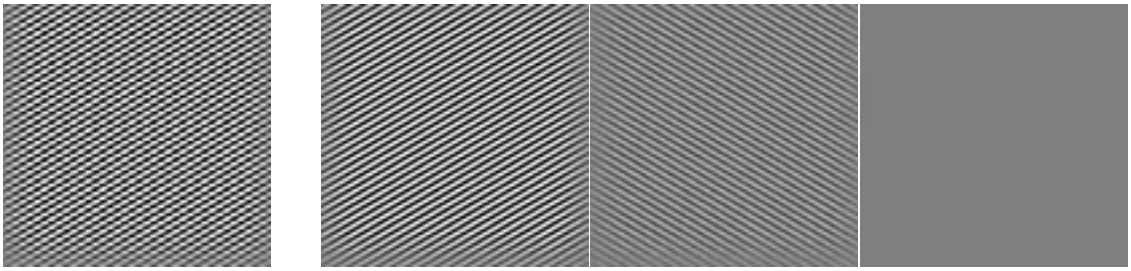
1. **Filtrage**

   (a) Voici trois images:



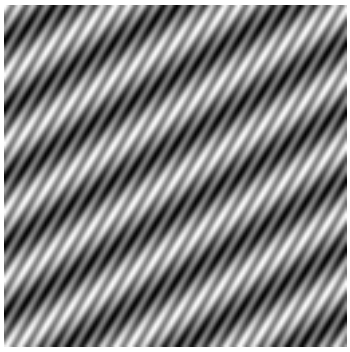
   Décrire (en la justifiant) l'opération de traitement d'image faite sur les deux premières images pour obtenir la troisième.
   <5 pts>

   (b) Dans la première image on voit une sorte de grillage avec des lignes diagonales droite et des lignes diagonales gauches. Dans les images qui suivent, seulement des lignes diagonales gauches dans la deuxième image et seulement des lignes diagonales droites dans la troisième et une image toute uniforme pour la quatrième. Quelles opérations de traitement d'image permet, à partir de la première image, de passer à la seconde puis à la troisième et enfin à la quatrième image.



   <5 pts>

   (c) Dans cette image, on voit une série de vagues (lesquelles sont vraiment (purement) sinusoïdale). L'image à été recalée entre 0 et 255 niveaux de gris. Expliquer de la façon la plus détaillée possible à quoi ressemblera le module du spectre de cette image de taille 128×128. Vous devez être capable de me dire exactement ce qui se trouve sur ce spectre (en module) au **pixel près** (en indiquant la **position** des pixel (ligne colonne) sur ce spectre de module où il y a de l'information spectrale).



<8 pts>

2. **Spectre**

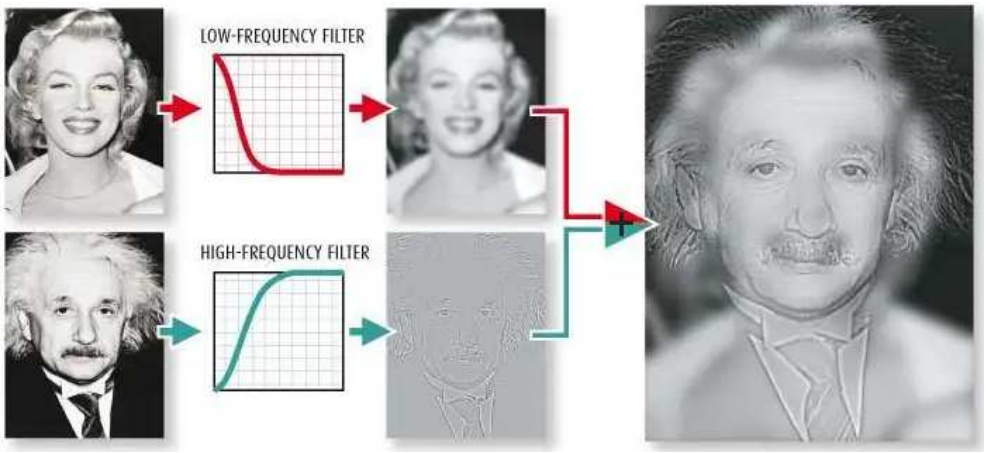
   (a) Soit le spectre d'une image de petite taille (disons 16×16), calculée par la FFT. Par quoi le spectre de cette *petite* image sera-t-il altéré si on le compare au spectre obtenu par sa version mathématique continue ? Que pourrait on faire pour atténuer cette dégradation ?
   <5 pts>

Réponse

1.(a)

On observe dans l'image finale les contours et les détails du visage (fréquences spatiales élevés) de A. Einstein mais aussi, en surimposition, et en plus floue, la forme et la silhouette du visage de M. Monroe. En fait, de près, les contours et détails du visage d'A. Einstein dominent, mais si on s'éloigne un peu, faisant disparaître ainsi les détails, les traits et l'expression fine et détaillée propre au visage d'A. Einstein, la silhouette floue et l'expression du visage de M. Monroe (zones homogène correspondantes aux fréquences spatiales basses) devient de plus en plus visible puis bientôt prépondérant, et prend finalement le dessus.

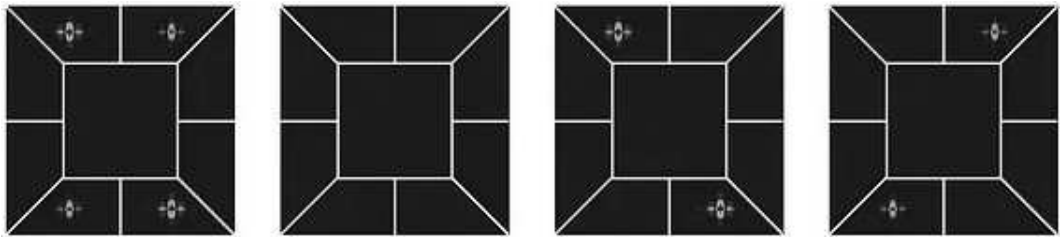
Les images étant déjà recalée au niveaux des yeux, du nez et de la bouche, un simple filtrage passe-bas de l'image de M. Monroe suivi d'un simple filtrage passe-haut de l'image du visage d'A. Einstein suivi d'une addition pixel par pixel des deux images (et un simple recalage des niveaux de gris finale entre 0 et 255) est sans aucun doute l'opération de traitement utilisée et demandée.
On peut dire aussi que puisque l'on ne voit pas de *ringing* (ou légère oscillation) sur l'image finale, on en déduit que le filtrage passe-haut et passe-bas ont une transition *smooth* dans le domaine fréquentiel (si ces derniers sont faits dans le domaine fréquentiel, ce qui est probablement le cas).
Finalement, avec les données que l'on nous donne, on ne peut être plus précis et donner une estimation de la fréquence de coupure de ces deux filtres. Donc, si on suppose que le filtrage est fréquentiel, on a l'opération de traitement suivante:



<5 pts>

1.(b)

À cause de la linéarité de la TF, Le spectre de l'image présentera 2 paires de pics de fréquences symétrique (et sur la même ligne du spectre du fait de la même fréquence) par rapport au centre du spectre. Une paire de pic est placée dans les hautes fréquences dans le quadrant droit du spectre (représentant les lignes lignes diagonales gauches du grillage) et l'autre dans le quadrant gauche du spectre (lignes diagonales droites) (cf. Figure ci-dessous sur laquelle on voit respectivement approximativement ce qui serait le spectre de l'image 1. 4. 2. et 3.). Pour séparer ces lignes selon leur direction, il suffit donc de faire un filtrage locale dans le quadrant approprié pour trouver la seconde ou troisième image puis un simple filtrage passe-bas pour trouver la quatrième image.
On ne peut être plus précis que cela (en donnant par exemple les fréquences numériques des filtres passe-bandes locales) car on ne nous donne pas la taille de l'image.



<5 pts>

1.(c)

On voit la surimposition (addition) de deux vagues sinusoïdales de différentes fréquences (une basse et l'autre plus élevée).

• La façon la plus simple de raisonner est la suivante. Intéressons nous à la vague de plus basse fréquence. On peut voir **3** périodes de cette vague pour la largeur (colonne) de cette image et **5** périodes de cette vague pour la longueur (ligne) de cette image. Il existera donc deux pics de Dirac symétriques par rapport au centre du module du spectre (recentré au centre de l'image) de cette image qui sont à la position:

$$\boxed{\text{Mat}[64+\mathbf{5}][64+\mathbf{3}]} \text{ et } \boxed{\text{Mat}[64-\mathbf{5}][64-\mathbf{3}]}$$

où Mat est la matrice du module (ou de la partie réelle du spectre, car il n'y a aucun déphasage des ondes dans cette image).

Pour l'onde de plus grande fréquence, on peut voir **25** périodes de cette vague pour la largeur (colonne) de cette image et **15** périodes de cette vague pour la longueur (ligne) de cette image. Il existera donc deux pics de Dirac symétriques par rapport au centre du module du spectre (recentré au centre de l'image) de cette image qui sont à la position:

$$\boxed{\text{Mat}[64+\mathbf{15}][64+\mathbf{25}]} \text{ et } \boxed{\text{Mat}[64-\mathbf{15}][64-\mathbf{25}]}$$

.

Notons aussi qu'en plus de ces 4 pics spectraux, il y aura aussi un pic spectral à la fréquence 0 correspondant à la valeur moyenne de l'image.

• La seconde façon de raisonner est très légèrement plus difficile mais a le mérite de préciser les fréquences numériques associées à ces deux ondes sinusoïdales.

La résolution fréquentielle de ce spectre est par définition $\Delta\nu = 1/NT$ où N est le nombre de pixel (qui est le même en longueur et largeur de cette image ($N = 128$) et T est la période de ces pixels. Puisque on ne sait pas quel est la valeur de T, on pose $T = 1$ et on se retrouvera avec des fréquences numériques et des périodes exprimées en nombre de pixels. Donc dans notre cas $\Delta\nu = 1/128$.

Intéressons nous à la vague de plus basse fréquence. On peut voir **3** périodes de cette vague pour la largeur (colonne) de cette image de taille 128 donc chaque période s'étale sur $\approx 128/3 = 42.6$ pixels ou

correspond à une fréquence numérique de $\approx 1/42.6 \approx 0.023$. Chaque pixel est séparé de $\Delta\nu = 1/128$ donc cette fréquence numérique correspond au $(1/42.6)/(1/128) = (3/128)/(1/128) = 3$ ième pixels (correspondant donc à la fréquence numérique en **colonne** de $\approx \boxed{0.023}$). Avec ce même raisonnement on trouve une fréquence numérique en **ligne** de $5/128 \approx \boxed{0.039}$ pour cette vague de basse fréquence.
Avec ce même raisonnement, on trouve pour la vague de plus haute fréquence une position au 15 ième pixel en ligne et 25 ième pixel en colonne pour une fréquence numérique de $\approx 15/128 = \boxed{0.117}$ en **ligne** et $\approx 25/128 = \boxed{0.195}$ en **colonne**.
A ces pics, il faut ajouter,bien sûr, leurs symétriques par rapport à 0 pour représenter les fréquences négatives (et la symétrie hermitienne du spectre).

<8 pts>

Nota: Il faut être **précis** quand on peut être précis et préciser ces fréquences numériques mathématiquement ou algorithmiquement (position des pixels associés aux pics spectraux existants). Dans le moins pire des cas, il faut faire un graphique pour préciser ou se trouveront approximativement ces 4 pics spectraux (en plus du pic centré sur la fréquence 0).

Notation: 3 points pour le raisonnement **qualitative** juste (*i.e.; ... 4 pics sur la diagonale droite; un faiblement espacé de l'origine et l'autre un peu plus espacé et symétrique deux à deux par rapport à l'origine plus le pic central calé sur la fréquence* 0, en expliquant à quoi ils correspondent respectivement sur l'image spatiale, etc... et 5 points pour le raisonnement **quantitatives**, pour les 4 mesures d'emplacement précis sur le spectre en pixels ou en fréquences numériques correspondantes.

2.(a)

Il faut se souvenir que le résultat de:
... La FFT (ou la TFD) (version discrète de la TF) d'une image est en fait équivalent à la Transformée de Fourier (TF) (mathématique, continue et parfaite !) qui s'appliquerait sur cette même image mais qui serait infiniment périodique en ligne et en colonne (ou qui serait cyclique ou toroïdale ou repliée sur elle même en ligne et en colonne)...
Du coup, plus l'image est petite et plus des éventuels discontinuités du signal image sur les bords de celle-ci (entre la première & dernière ligne et la première & dernière colonne) induit d'éventuelles discontinuités et donc de fausses hautes fréquences qui sont des fréquences parasites artificielles lesquelles ne font pas parties du vrai spectre (mathématique et continue) de l'image. Pour une image 16×16, ces discontinuités représentent une partie importante de l'image puisqu'ils concernent 25% (de l'ensemble des pixels) de l'image $((16 \times 4)/(16 \times 16))$.

Pour annuler ces dégradations, on peut, par exemple, *pariser* (rendre pair) l'image (grâce à une symétrie miroir 2D) ou faire une DCT (Transformée en cosinus discret) qui rend l'image, par ce traitement, implicitement paire.

<5 pts>

III. Restoration (16 pts)

Vous travaillez sur une image astronomique de taille 512×512 pixels obtenue par un télescope au sol. En raison des turbulences atmosphériques, l'image est floue et on sait que la PSF, due à ces turbulences atmosphériques, a une forme Gaussienne (de paramètre σ). La PSF varie spatiale ment en raison des turbulences atmosphériques, qui élargissent la PSF de manière à travers le champ de vision. Sur l'image de taille 512×512 pixels, la PSF est plus étroite au centre de l'image et est de $\sigma = 2$ pixels puis s'élargit linéairement (selon la distance par rapport au centre) sur les bords pour être maximale sur les 4 points de l'image les plus éloignés du centre pour être de $\sigma = 4$ pixels.

1. On aimerait faire une image synthétique pour simuler ce floue. Pour cela on dispose d'une image étoilée sans turbulence atmosphérique sur laquelle on désire recréer cette dégradation. Écrire la PSF et dire comment vous allez procéder ensuite avec cette PSF (spatialement, fréquentiellement, etc.) en motivant vos raisons.
   <7 pts>

2. Connaissant cette PSF, peut-on utiliser le filtre de Wiener pour procéder à la déconvolution d'une image astronomique dégradée par ce type de floue ? Si la réponse est non, expliquer précisément pourquoi. Peut-on utiliser la méthode de Landweber ou Tichonov-Miller ou celle vue en TP pour procéder à la déconvolution ? Si la réponse est oui expliquer ce qui devrait être changer dans la méthode que vous avez programmé en TP (en plus du fait que, bien sur, la PSF est différente et variante spatialement).
   <5 pts>

3. Donner des exemples (d'images) dans lesquels le flou varie spatialement linéairement d'un point de l'image à un autre.
   <4 pts>

Réponse

1.

La PSF bidimensionnelle ($\mathrm{PSF}(\sigma; x, y)$), dont l'écart type varie spatialement dans l'image de taille 512×512 pixels (size × size pixels), s'écrit:

▷ $$\frac{\exp\left\{-\frac{(x-x_o)^2+(y-y_o)^2}{2\,\sigma^2(x,y)}\right\}}{2\,\pi\cdot\sigma^2(x,y)} \quad \text{avec: } \sigma(x,y)=2+2\cdot\frac{\sqrt{(x-x_o)^2+(y-y_o)^2}}{\text{size}/2} \quad \text{et: } x_o=y_o=256=(\text{size}/2)$$

En fait, on peut oublier la constante de normalisation 2π au dénominateur si on prend garde, qu'une fois la forme de cette PSF de ce filtre passe-bas créée sous forme d'image discrète, avec la forme:

▷ $$\exp\left\{-(x-x_o)^2+(y-y_o)^2/2\,\sigma^2(x,y)\right\}/\sigma^2(x,y),$$

il suffira ensuite de normaliser cette image à un (les coefficients de cette PSF doivent sommer à un), pour faire en sorte que la moyenne de l'image floue soit la même que la moyenne de l'image non floue (comme tout filtre passe-bas).

Soit $I(x, y)$, l'image étoilée sans turbulence et $I_d(x, y)$, l'image avec la dégradation de floue, on devra réaliser (avec possiblement un bruit additif $n(x, y)$):

$$I_d(x, y) = I(x, y) * \mathrm{PSF}(x, y) + n(x, y)$$

Comme la PSF est variante spatialement dans l'image, on est obligé de procéder dans le domaine spatial et pas dans le domaine fréquentiel pour réaliser cette convolution (qui est une opération appliquée *globalement* et indifféremment sur toute l'image).
<7 pts>

2.

Non le filtre de Wiener ne peut être utilisé car on ne peut faire une convolution dans le domaine spectral (lequel ne peux utiliser qu'une PSF qui *globalement* est la même sur toute l'image) avec une PSF qui varie spatialement. On peut utiliser la la méthode de Landweber ou Tichonov-Miller ou celle vue en TP pour procéder à la déconvolution mais dans ce cas, il faut s'assurer que toutes les convolutions se fassent spatialement et non plus fréquentiellement, ce qui nous permettra d'utiliser notre PSF variant spatialement. Algorithmiquement, à chaque position (x, y) de la convolution, il suffit de calculer localement $\sigma(x, y)$ puis utiliser $\mathrm{PSF}(\sigma; x, y)$ (variant avec (x, y); cf. Question 1.).
<5 pts>

3.

• Une image d'une autoroute vue de face; le flou variant spatialement et est plus petit, plus on s'éloigne de la caméra (car la voiture semble se déplacer moins vite).

• Une image capturée à travers une atmosphère avec un gradient linéaire de densité ou de contenu en aérosols ou poussières ou pollution (par exemple, une transition entre une zone claire et une zone brumeuse). Les détails (comme les routes ou les bâtiments) sont nets dans la région claire, mais deviennent progressivement d'autant plus flous qu'ils sont éloigné à mesure que la densité atmosphérique augmente.

• Image sous marine. Plus les objets sont éloignés et plus la densité du volume d'eau entre eux et la caméra est grand et induit un flou important.

• Image avec mouvement de caméra (*hand-shaking*).

• Image avec turbulence atmosphérique (plus les objets sont loin à travers cette turbulence atmosphérique et plus la flou semble important).

• Image avec plusieurs plans et mise au point uniquement localisée pour le premier plan.

• ...

<4 pts>

IV. Codage en C (33 pts)

Soit le modèle de dégradation linéaire suivant: $y = x + n$ dans lequel x représente une image (de départ) naturelle en 256 niveaux de gris de taille (length,width) et y, sa version bruitée par du bruit (noté ici par n) additif blanc Gaussien (AWGN).
Dans ces images, $y(i,j)$, $x(i,j)$ et $n(i,j)$ correspondent respectivement au niveau du gris du pixel de coordonné ligne$= i$ et colonne$= j$ (ou au site $s = (i,j)$) de l'image bruitée et de l'image originale de départ (non bruitée) $x(i,j)$. $n(i,j)$, quand à lui est la valeur du bruit additif Gaussien qui dégrade cette image x au pixel de même coordonné (ligne$= i$ et colonne$= j$).
On aimerait retrouver l'image x (*i.e.,* l'image initiale et sans bruit) à partir de y (l'image bruitée qui est la seule observable) par une procédure de minimisation (locale) de type ICM (*Iterated Conditional Modes*). Plus précisément, on aimerait trouver x, par ICM, qui minimise la fonction d'énergie globale: $U(x,y) = \|y - x\|_2^2 + \text{Total-Variation}(x)$ (si on considère x et y comme un vecteur de pixels) ou que l'on peut détailler plus précisément comme suit:

$$U(x,y) = \sum_{s \in S} \left\{ [y_s - x_s]^2 + \beta \cdot \overbrace{\sum_{s \in \eta_s} |x_s - x_{\eta_s}|}^{\text{Total-Variation}(x_s)} \right\} \tag{1}$$

S, représente tous les sites (ou pixels) de l'image, et η_s représente un voisinage du second ordre (*i.e.*, les 8 plus proche voisins [ppv] du pixel au site s).
Le deuxième terme, qui est sommé sur tous les site s, est la totale variation de x et peut être vu comme la somme de la norme 1 des gradients au site s dans toutes les directions (dans cet exemple pour les 8 ppv de s). Ce terme est généralement assez faible pour une image non bruitée et d'autant plus élevé pour une image qui est d'autant plus bruitée par du bruit Gaussien. La recherche de x, qui minimise $U(x,y)$ par l'ICM recherchera donc une solution image (en niveaux de gris) qui n'est pas trop loin au sens des moindres carrées) de l'image bruitée mais qui possédera une totale variation beaucoup plus faible, *i.e.* correspondant à une image débruitée.

Inspirez vous du TP numéro 5 Segmentation Markovienne 2 classes par ICM. Je rappelle que l'ICM que vous avez utilisez en TP pour trouver l'image segmentée x à partir d'une image (possiblement bruitée) y minimisait la fonction d'énergie globale:

$$U(x,y) = \sum_{s \in S} \left\{ \left[\ln(\sqrt{2\pi}\sigma_{x_s}) + \frac{(y_s - \mu_{x_s})^2}{2\sigma_{x_s}^2} \right] + \beta_c \cdot \sum_{s \in \eta_s} \left[1 - \delta(x_s, x_{\eta_s}) \right] \right\} \tag{2}$$

dans laquelle: $y_s = y(i,j)$ représentait le niveau du gris du pixel, de coordonné ligne$= i$ et colonne$= j$ (ou au site $s = (i,j)$), de l'image originale (que l'on désire segmenter) en niveaux de gris, η_s était les 4 ppv de s et $(\mu_{x_s}, \sigma_{x_s})$, représentait la moy. et la var. de (la distrib. Gaussienne associée à) la classe x_s ($\in \{e_0, e_1\}$).

1. Considérons que l'image à débruiter (de longueur et largeur $lgth$ et $wdth$ est chargée dan le tableau de flottant $ImgY$ et que l'on a déjà alloué de la mémoire pour une image (tableau 2D) de même taille pour $ImgX$ (l'image débruitée que l'on cherche).
   Donner le code en C (essentiellement la boucle principale) de cette procédure de débruitage par ICM.
   <25 pts>

2. Expliquez le rôle de β dans cette procédure.
   <4 pts>

3. En vous inspirant des connaissances apprises lors du TP5, expliquer et/ou rappeler comment on pourrait algorithmiquement s'assurer que votre procédure de débruitage par ICM fonctionne bien et a été probablement bien programmé.
   <4 pts>

Réponse

1.

Dans la fonction d'énergie donnée par l'Équation (1), on reconnaît que le premier terme: $[y_s - x_s]^2$ est en fait le terme d'attache aux données, lequel ressemble pas mal au terme d'attache aux données de l'énergie Markovienne associée à notre problème de segmentation (cf. Eq. (2)) et le deuxième terme: $\sum_{s\in\eta_s} |x_s - x_{\eta_s}|$, quand à lui, est le terme d'énergie *a priori* pour lequel, en fait, si on le compare au terme d'énergie *a priori* de notre énergie Markovienne pour la segmentation (cf. Eq. (2)) est un terme qui *caractérise et modélise les propriétés de l'image $\hat{x}$ que l'on débruitée que l'on cherche à estimer (dans laquelle on recherche des zones de niveaux de gris qui sont smooth ou lisses).* On peut facilement modéliser ce terme en disant que $\hat{x}$ est une réalisation d'un champ Markovien associé a un voisinage du second ordre utilisant des cliques binaires (horizontales et verticales), comme pour le problème de la segmentation, mais avec des potentiels de cliques du style cette fois de: $V_c(x) = |x_s - x_{\eta_s}|$ (ou $V_c(x) = |x_s - x_t|$, pour reprendre les même notations que celle du cours). Du coup, on peut appliquer, pour minimiser cette fonction d'énergie globale, l'algorithme d'optimisation itératif ICM (qui le fera avec une complexité linéaire (aux nombre d'itérations près)).

L'image en niveaux de gris $ImgX$ que l'on cherche peut être vu comme une image avec "256" classes (de niveaux de gris). En considérant cela, on doit faire comme pour le TP5, *i.e.*, exprimer cette énergie globale $U(x, y)$ localement pour chaque pixel (*i.e.*, sans le signe $\sum_{s\in S}$ dans l'Eq. (1)), associer ensuite l'énergie locale pour les 256 niveaux de classe de gris possible et prendre pour ce pixel la classe de niveaux de gris assurant l'énergie la plus faible et recommencer pour chaque pixel et pour différentes itérations (jusqu'à stabilité de cette procédure d'ICM qui est, ni plus ni moins, qu'une procédure de descente de gradient, à direction alternée et pas fixe).

Dans cette procédure (cf. page suivante), j'ai mis ce qui n'était pas demandé dans l'énoncé mais qui est **indispensable** pour cet algorithme. À savoir, le calcul et l'affichage de l'énergie globale à chaque itération, l'arrêt si on a convergence de la procédure; DiffMap(ImgX,ImgTmp,lgth,wdth) étant une fonction qui calcule la différence en nombre de pixels des niveaux de gris entre les deux matrices $ImgX$ et $ImgTmp$.

Notation:
Les cinq éléments indispensables à ce programme sont (et chacun noté 5 points) sont:
▷ **1.** Pour tous les pixels (i, j) de l'image X initialement remplie par Y.
▷ **2.** Calcul de (la bonne) énergie locale (**avec les 8 ppv !**) **pour les** 256 **niveaux de gris possibles** pour ce pixel.
▷ **3.** Sélection du niveau de gris associée à l'énergie la plus faible.
▷ **4.** Ajout de cette énergie locale pour le calcul de l'énergie finale (que l'on **doit imprimer** à chaque itération.
▷ **5.** Après convergence (plus aucun changement de X (ou au pire après un max. d'itérations); arrêt de la procédure.

```
. int i,j,k,l;
. int LabelNrjMin,NbPixDif;
. float NrjTotVar,NrjMin,SumNrj;
. float** ImgTmp=fmatrix_allocate_2d(lgth,wdth);
. float* VctNrj=fmatrix_allocate_1d(255);
. for(k=0; k<255; k++) VctNrj[k]=0.0;
. for(i=0; i<lgth; i++) for(j=0; j<wdth; j++) ImgX[i][j]=ImgY[i][j];
.   //>ICM_Loop
.   for(k=0; k<ITER_MAX; k++)
.   { SumNrj=0.0;
.     for(i=0; i<lgth ; i++) for(j=0; j<wdth; j++) ImgTmp[i][j]=ImgX[i][j];
.     for(i=1; i<(lgth-1) ;i++) for(j=1; j<(wdth-1); j++)
.         { for(l=0; l<255; l++) VctNrj[l]=100000;
.           for(l=0; l<=255; l++)
.             { NrjTotVar=0.0;
.               NrjTotVar+=fabs(ImgX[i-1][j]-(float)l);
.               NrjTotVar+=fabs(ImgX[i-1][j-1]-(float)l);
.               NrjTotVar+=fabs(ImgX[i][j-1]-(float)l);
.               NrjTotVar+=fabs(ImgX[i+1][j-1]-(float)l);
.               NrjTotVar+=fabs(ImgX[i+1][j]-(float)l);
.               NrjTotVar+=fabs(ImgX[i+1][j+1]-(float)l);
.               NrjTotVar+=fabs(ImgX[i][j+1]-(float)l);
.               NrjTotVar+=fabs(ImgX[i-1][j+1]-(float)l);
.               VctNrj[l]=CARRE(l-ImgY[i][j])+Beta*NrjTotVar; }
.           NrjMin=100000;
.           LabelNrjMin=-1;
.           for(l=0; l<255; l++) if (VctNrj[l]<NrjMin) { NrjMin=VctNrj[l]; LabelNrjMin=l; }
.           if (LabelNrjMin!=-1) ImgX[i][j]=LabelNrjMin;
.           SumNrj+=VctNrj[LabelNrjMin];
.         }
.     NbPixDif=DiffMap(ImgX,ImgTmp,lgth,wdth);
.     printf("\n iter[%d] Nrj=%.0f [Diff=%d]",k,SumNrj,NbPixDif);
.     if (!NbPixDif) break;
.   }
```

<25 pts>

Nota: Une variante possible de cette procédure serait de calculer le niveau de gris en chaque pixel (qui assure le minimum de la fonction locale d'énergie) pour le niveaux de gris courant, le niveau de gris $+1$ et le niveau de gris -1 et prendre pour ce pixel la classe de niveaux de gris assurant l'énergie la plus faible et recommencer pour chaque pixel et pour différentes itérations (jusqu'à stabilité). Cette procédure demanderait moins de calcul par itération mais plus d'itérations et serait un peu moins efficace mais je la compte bonne quand même.

2.

β est le paramètre de régularisation qui va pondérer l'énergie dit de vraisemblance (premier terme de la fonction d'énergie) avec le terme d'énergie *a priori* (de totale variation). Plus l'image est bruitée, plus ce terme doit être important. Voir aussi ce qui est dit dans la question suivante.

<4 pts>

3.

• Pour $\beta = 0$, la convergence de l'algorithme se fera en une itération et l'algorithme ne fait rien; on retrouve l'image bruitée.

• Pour $\beta \neq 0$, la fonction d'énergie globale, à chaque itération, doit constamment diminuer sans oscillation et sans remontée d'énergie. Le nombre de changement de niveaux de gris entre la matrice de niveaux de gris à l'itération p et $p+1$ devrait aussi tendre à diminuer globalement (un peu comme une sorte de décroissance en exponentielle décroissante) jusqu'à une certaine **stabilité** pour aucun (ou très peu) de changement vers les dernières itérations.

• Plus β est important, plus le terme de régularisation est important dans cette fonction d'énergie à minimiser

et plus on cherchera à minimiser prioritairement la totale variation de l'image débruitée, *i.e.*,, sa variation de niveaux de gris (au sens de la norme 1), favorisant ainsi la présence de zones de régions homogènes. Attention; cela ne veut pas dire que plus β est important, plus l'image sera débruitée, cela veut dire que plus β est important, plus l'image débruitée obtenue montrera des zones de régions homogènes.
• Pour β très très grand, le premier terme de la fonction d'énergie $\sum_{s\in S}[y_s - x_s]^2$ devient négligeable et le deuxième terme (Variation Totale) devient prépondérant ($\sum_{s\in S}\sum_{s\in\eta_s}|x_s - x_{\eta_s}|$) dans le cadre de cette minimisation; l'image débruitée tendra à converger vers une image avec beaucoup de zones de niveaux de gris uniformes et tendre pour de très très grande valeur de β vers une image quasi uniforme (qui serait la valeur moyenne de l'image et qui serait associé à une variation totale quasi-nulle).
<4 pts>

Nota: Voici un exemple de débruitage donnée par cette algorithme (pour une image bruitée par du AWGN de $\sigma = 30$). Le code est disponible sur la page Web du cours à coté de cette correction.

Image bruitée avec $\sigma = 30$ et débruitée avec cet algorithme ($\beta = 30$)

VI. Misc. & Questions Relatives aux TPs (17 pts)

1. TP numéro 5 Segmentation Markovienne 2 classes par Recuit Simulé.
   Dans le TP5, pour la dernière question D. (Segmentation Markovienne supervisée avec Recuit Simulé), je vous ai donné, pour vous aider à coder, une fonction en langage C qui s'appelait:
   float choose_label_sa(float* prob) permettant de renvoyer une classe parmi deux classes possibles (classe "0" ou classe "1") selon la densité de probabilité discrète donnée par le vecteur prob de longueur 2 (variable binomiale pour une segmentation deux classes).
   Je vous demande de me donner maintenant le code en langage C de la fonction:
   int choose_3labels_sa(float* prob) permettant de faire la même chose mais pour trois étiquettes de classes dont l'occurrence serait donnée par la densité de probabilité prob de longueur 3 pour une variable trinomiale qui serait donc utilisée dans le cadre d'une procédure de Recuit Simulé (mais aussi dans toutes autres procédures de segmentation stochastique trois classes).
   <5 pts>

2. TP numéro 5 Segmentation Markovienne 2 classes par ICM.

(a) L'algorithme des K-moyennes va permettre de rassembler en cluster (plus précisément en boules sphériques [de dimension n]) différents vecteurs de caractéristiques n-multidimensionnel . Cependant si ses vecteurs sont des attributs de textures ou de simples niveaux de gris, chacun des vecteurs (associés donc à un pixel de l'image), cet algorithme ne prendra pas en compte le fait que les pixels voisins ont plus de chance d'appartenir à la même classe (ou au même cluster) comme le ferait une segmentation Bayesienne-Markovienne. Comment pourrait-on modifier l'algorithme des K-moyennes pour qu'il prenne en compte cet information contextuelle *a priori* ? Proposer une ou plusieurs suggestions.
<5 pts>

(b) Pour des problèmes de classification impliquant un nombre de caractéristiques importants, (i.e., de grande dimensionnalité), l'algorithme des K-moyennes convergera vers différentes partitions ou différents groupements (correspondant à un minimum local) suivant la position initiale des K centres donnée à la première itération de l'algorithme.
Imaginez que vous ayez un problème de classification de grande dimensionnalité, et un algorithme des K-moyennes qui choisira la position initiale de ces K centres de façon totalement aléatoire. vous voulez créer un partitionnement des données à l'aide de cet algorithme, mais le lancement de dix fois cet algorithme des K-moyennes vous donnera 10 partitionnements différents.
Quelle serait la procédure la plus intelligente qui permettrait de sélectionner le partitionnement parmi les 10 qui serait le "meilleur" ? Discuter aussi les différents sens que pourrait signifier le mot meilleur ?
<5 pts>

3. À partir d'une image segmentée ou image de classe issue d'une segmentation, on aimerait faire, pour les besoins d'un algorithme, une image de classe 8 fois plus petite. Donner le pseudo code ou dire quelles serait votre stratégie pour le faire.
<2 pts>

Réponse

1.

Les probabilités; PROB[0], PROB[1] et PROB[2] définiront trois intervalles dont la longueur est proportionnelle à la probabilité d'occurrence de chacune de ces trois classes et dont la somme est de longueur 1. La longueur du premier intervalle est PROB[0], la longueur du deuxième intervalle est la longueur comprise entre PROB[0] et (PROB[0]+PROB[1]) et la longueur du troisième intervalle est comprise entre (PROB[0]+PROB[1]) et 1. Il nous reste plus qu'à lancer un nombre aléatoire de distribution uniforme entre 0 et 1 et voir où ce dernier tombera comme avec une roue de loterie. Plus l'intervalle considéré associé à une classe est grand et plus ce nombre aléatoire aura une chance importante de tomber sur l'intervalle de cette classe.

```
. int choose_3labels_sa(float* prob)
.   {
.     float RandValBetw_Zer_One = ((float)rand()/RAND_MAX)
.     if (RandValBetw_Zer_One < prob[0])              return 0;
.     if (RandValBetw_Zer_One < (prob[0]+prob[1])) return 1;
.     if (RandValBetw_Zer_One < 1.0)                  return 2;
.   }
```

La dernière ligne pourrait être remplacée simplement par RETURN 2; seulement.
<5 pts>

2.(a)

On pourrait ajouter dans le vecteurs de caractéristiques multidimensionnel la position x et y de chaque pixel (ou sa position log-polaire (r, θ) par rapport au centre de l'image). Dans ce cas, l'algorithme va chercher à grouper des pixels proches en niveaux de gris **et** aussi en position sur la grille rectangulaire de l'image. On peut aussi prendre l'ensemble des niveaux de gris (ou de caractéristiques de textures) sur une petite fenêtre (par ex. $8x8$) centrée sur un pixel ou alors en post-traitement, filtrer simplement le résultat (l'image partitionnée/segmentée) par un filtre médian.
<5 pts>

2.(b)

Le but de l'algorithme des K-moyennes est de minimiser un critère qui est la somme des variances intraclasses de chaque échantillon à classer. Autrement dit, ce critère représente concrètement la somme, pour les K clusters (ou classe), de la dispersion (ou variance) de chaque échantillon autour de leur centre respectif. Une solution pour trouver quel serait le "meilleur" partitionnement serait pour chacune des solutions de calculer ce critère et de sélectionner la partition (ou solution) associée à la dispersion la plus faible. Dans ce cas le mot meilleure signifierait, la solution qui est la plus intéressante au sens du critère de dispersion de l'algorithme des K-moyennes (ceci doit être fait bien sûr pour un K donné/fixé car cette dispersion va continuellement diminuer pour un K croissant). Dans le cas de l'utilisation du K-moyennes, dans le cadre de l'estimation des paramètres d'un mélange de distribution, on pourrait prendre la partition qui maximise sa vraisemblance $\sum_{s \in S} V(x_s, y_s) = \sum_{s \in S} \left[\ln(\sqrt{2\pi}\sigma_{x_s}) + (y_s - \mu_{x_s})^2 / 2\sigma_{x_s}^2\right]$ (Chap. Segmentation d'images, slide [16]).

On pourrait définir un autre critère qui serait associer à la qualité de la segmentation ou partitionnement obtenu pour ce que l'on veut ensuite en faire. Dans ce cas le mot *meilleur* aurait une signification différente et serait lié a l'application haut niveau qui utiliserait cette segmentation. Par exemple, si on utilise cette segmentation pour une application (de haut niveau) de reconnaissance des formes, on pourrait utiliser l'algorithme de RdF sur les 10 segmentations obtenues et retenir la solution de RdF qui serait celle associée à la solution de reconnaissance du classifieur RdF qui serait la moins ambiguë ou la plus fiable.
<5 pts>

3.

Pour tous les carrées de taille 8×8, on trouverait la classe majoritaire et on donnerait cette classe au pixel concerné dans l'image réduite.
<2 pts>