ENSTA PARIS

MIV308 : RECONNAISSANCE DE FORMES

TP 3 : Distance de visibilité météorologique et restauration de la visibilité

Réalisé par :
Amin HENTETI

Enseignants :
M. Jean-Philippe Tarel
M. IENG Sio-Song

Année universitaire: 2019/2020

1 Introduction

Le but de ce TP est la détection du brouillard par l'estimation de la distance de visibilité. Dans la première partie de ce TP, on extrait un profil vertical en intensité. Puis on fait le calcule le point d'inflexion de ce profil. Dés alors on déduit la distance de visibilité Enfin on restaure la visibilté dans une image et teste cette méthode.

Pour chaque question dans ce compte rendu, j'ai ajouté le code de la réponse qui correspond au question. Et dans certains question j'ajoute des résultats graphique nécessaires. De plus le code de Matlab est sous forme de script structuré par des blocs et pour chaque de question il en suit le code de la réponse. J'ai décomposé le code en deux parties vu que la dernière partie : Restauration de la visibilité dans une image est indépendant des autres. Et ceci permet de réduire le nombre des lignes de code dans un seul fichier.

2 Profil vertical des intensités

La loi de Koschmieder donne l'atténuation d'un objet d'intensité I_0 lorsqu'il est observé à travers un brouillard de coefficient d'extinction k à une distance d. L'intensité vue de l'objet est:

$$I(d) = I_0 e^{-kd} + I_s(1 - e^{-kd}) \tag{1}$$

où I_s est l'intensité du ciel. Grâce à cette loi, nous pouvons obtenir l'équation de la variation de l'intensité d'une zone plane de couleur et texture uniforme vue en perspective, lorsqu'il y a du brouillard. Soit I_0 l'intensité de cette surface. La surface est supposée observée comme étant au sol avec une ligne d'horizon placée à la hauteur v_h dans l'image. La relation qui lie la distance d d'un point sur la surface au sol et le numéro de ligne v de l'image de ce point est:

$$d = \frac{\lambda}{v - v_h} \tag{2}$$

où λ est une combinaison des paramètres intrinsèques et extrinsèques de la caméra.

Travail à effectuer

Question 1

En substituant l'équation (2) dans (1), on obtient le modèle théorique du profil vertical de variation de l'intensité due au brouillard :

$$I(v) = I_0 e^{-\frac{k\lambda}{v-v_h}} + I_s(1 - e^{-\frac{k\lambda}{v-v_h}}) = I_s + (I_0 - I_s)e^{-\frac{k\lambda}{v-v_h}} \tag{3}$$

Question 2

On trace le le modèle théorique $I(v)$ pour $v \in [1, 20]$ avec les données en énoncé $v_h = 0, \lambda = 500, k = 0.02, I_s = 1.0$ et $I_0 = 0.2$. La figure **??** représente ce modèle.

```
v =[1:0.05:20]; vh = 0; lambda = 500;
k = 0.02; Is = 1.0; I0 = 0.2;
I = I0*exp(-k*lambda./(v-vh)) +Is*(1-exp(-k*lambda./(v-vh)));
figure('Name',' modele theorique I(v) ');
plot(v,I);
xlabel('v'); ylabel('I');
```

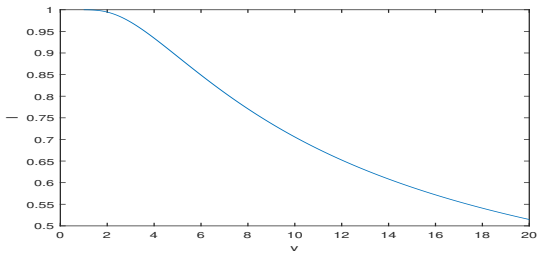


Figure 1: modèle théorique $I(v)$

Question 3

On cherche la point d'inflexion en dérivons l'équation (3) deux fois par rapport à v et on cherche où il s'annule :

$$I^{'}(v) = -k\lambda\,(I_0 - I_s)\left(\frac{-1}{(v-v_h)^2}\right)e^{-\frac{k\lambda}{v-v_h}}$$

$$\Longrightarrow I^{''}(v) = -k\lambda\,(I_0 - I_s)\left(\frac{2}{(v-v_h)^3} - \frac{k\lambda}{(v-v_h)^4}\right)e^{-\frac{k\lambda}{v-v_h}}$$

$$I^{''}(v_i) = 0 \Longleftrightarrow \frac{2}{(v_i-v_h)^3} = \frac{k\lambda}{(v_i-v_h)^4} \Longleftrightarrow \quad k = 2\frac{v_i - v_h}{\lambda} \quad (4)$$

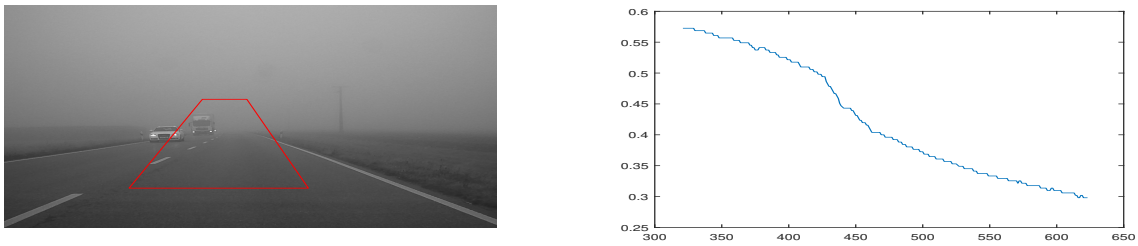
avec v_i est la hauteur du point d'inflexion.

Question 4

On stocke l'image que l'on dispose dans une variable **imad** au format double tout en normalisant. Et on exécute la fonction **profil** donné dans l'énoncé.

```
ima = imread('F01.png');
imad = double(ima)/255;
data = profil(imad);
figure('Name','data');
plot(data(:,1),data(:,2));
```

En choisissant 4 points qui forme un trapèze on peut sélectionner une partie de l'image comme montré dans la figure 2(a). On utilise la matrice retourné par cette fonction qui est avec un nombre de ligne égal aux différences des hauteurs maximum et minimale du trapèze ou parallélogramme sélectionné et de 2 colonnes. On plot alors dans la figure 2(b) la deuxième colonne du résultat en fonction de la première pour avoir le profil d'intensité de cette zone qui est de même allure que le modelé théorique.



(a) zone sélectionné

(b) profil d'intensité

Figure 2: Exécution de la fonction **profil**

Question 5

On sélectionne trois zones différentes sur la voie de gauche, centrale et une zone qui chevauche la route et le champ. On affiche le profil correspondant pour chaque zone.

```
zone1 = profil(imad);
zone2 = profil(imad);
zone3 = profil(imad);
figure('Name','zones');
plot(zone1(:,1),zone1(:,2));
hold on;
plot(zone2(:,1),zone2(:,2));
hold on;
plot(zone3(:,1),zone3(:,2));
legend('voie de gauche','voie centrale','chevauchement route et champs');
```

(a) voie de gauche (b) voie centrale (c) chevauchement route et champs

Figure 3

D'après la figure 4 des 3 zones tracés dans la figure 3, la zone centrale présente le profil le plus adéquat à traiter puisqu'il est le plus lisse et proche du modèle théorique.

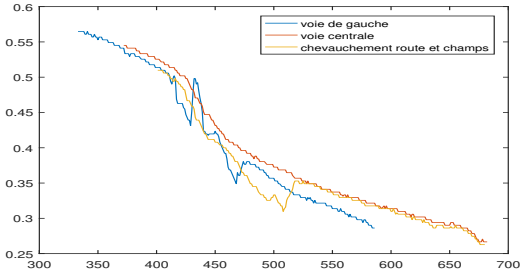


Figure 4: Le profil vertical des

3 Distance de visibilité

La position du point d'inflexion peut aussi s'obtenir comme le lieu où la dérivée première a un extrémum local. Après avoir déterminé la position du point d'inflexion du profil vertical choisi, on peut déduire le coefficient d'extinction k par la formule du question 3. A partir de cette dernière valeur, on calcule la distance de visibilité en mètres, définie comme la distance à laquelle le contraste d'un objet noir devient inférieur à 5% par la formule : $d_{visib} = \frac{3}{k}$ (5)

Travail à effectuer

Comme expliqué dans la question précédente je vais m'intéresser sur le profil de la zone centrale qui est stocké dans la variable **zone2** ; c'est pourquoi j'utilise le numéro 2 dans les nouveaux noms des variables.

Question 1

On approxime le dérivé premier du profil par la méthode de différence finis $f^{'}(x) = f(x+1) - f(x)$

```
dzone2 = 0*zone2;
dzone2(:,1) = zone2(:,1);
dzone2(1:end-1,2) = zone2(2:end,2)-zone2(1:end-1,2);
figure('Name','derive profil zone centrale');
plot(dzone2(:,1),dzone2(:,2));
```

En affichant dans la figure 5 le résultat de cette approximation du dérivé, on remarque plusieurs points nuls et des pics aléatoires. Ceci est due aux forme escalier du profil.

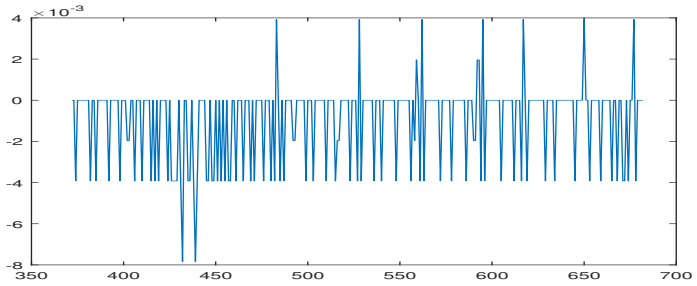


Figure 5: dérivé première du profil par difference finis avec pas = 1

Question 2

Afin d'éviter la forme escalier du profil, on fait le lissage avec un filtre moyen de taille 1*21 (le profil est un vecteur 1D). On dispose des vecteurs 1D donc on peut utiliser la fonction **conv** qui calcule la convolution ou encore le filtrage. On s'intéresse aux gestions des bords vue que la convolution ne donne pas des valeurs "cohérents" aux deux frontières du vecteur résultat puisque cette opération fait comme s'il y a des valeurs nulles au-delà de ces frontières. On peut prend on compte les frontières en ajoutant des bouts de bords aux frontières du profil avant la convolution et puis après convolution on les ignore. Remarque : J'emploie une technique de **slicing** pour inverser l'ordre des colonnes dans une matrice voici un exemple qui explique cette opération :

```
>> l=[1 2 3 4;5 6 7 8]
l =

     1     2     3     4
     5     6     7     8
>> l(:,end:-1:1)
ans =
     4     3     2     1
     8     7     6     5
```

Cette méthode sera utilisée ici avec un vecteur pour étendre ses frontières. Il permet une gestion des bords afin d'avoir une sorte continuité entre les pixels adjacents des bords et des bordures ajoutées.

```
bordure1 = zone2(1:len_fil,2);
bordure1 = bordure1(end:-1:1);
bordure2 = zone2(end-len_fil:end,2);
bordure2 = bordure2(end:-1:1);
extend_boundary = [bordure1; zone2(:,2); bordure2];
figure('Name','boundaries');
plot(zone2(:,1),zone2(:,2),'*');
hold on
x1 = zone2(1:len_fil,1)-len_fil+1;
x2 = zone2(:,1);
x3 = zone2(end-len_fil:end,1)+len_fil;
extend_x_axis = [x1;x2;x3];
plot(extend_x_axis , extend_boundary);
legend('normal','avec bordure');
```

On peut voir dans la figure 6 la version étendue du vecteur qui contient le profil d'une zone centrale sélectionné. Avec ce vecteur étendu on fait la convolution avec un filtre moyenne de taille 21 avec l'option **same** pour avoir la même taille que le premier vecteur en entré de la fonction **conv** puis on extrait le vecteur sans les bordures ajoutés.

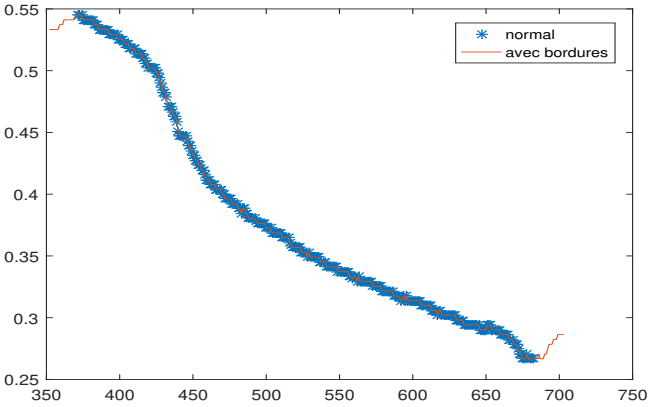


Figure 6: Version étendue du vecteur profil

```
fil_moy = fspecial('average',[1 21]);
len_fil = length(fil_moy);
result = 0*zone2;
result(:,1) = zone2(:,1);
result_with_boundary = conv(extend_boundary,fil_moy,'same');
result(:,2)= result_with_boundary(len_fil+1:end-len_fil-1);
figure('Name','lissing');
plot(result(:,1),result(:,2));
hold on;
plot(zone2(:,1),zone2(:,2));
legend('lisse','original');
```

D'après le résultat de lissage donné dans la figure 7, on remarque que le profil ne s'est pas très varié et alors le point d'inflexion ne s'est pas trop déplacé.

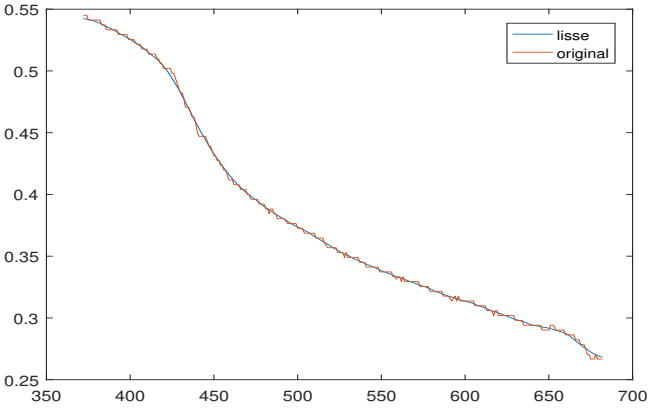


Figure 7: lissage du profil par un filtrage moyen

Question 3

On calcule comme précédemment la dérivée du profil lissé et on cherche la position de son minimum. On affiche le point d'inflexion calculé dans la figure 8.

```
dresult = 0*zone2;
dresult(:,1) = zone2(:,1);
dresult(1:end-1,2) = result(2:end,2)-result(1:end-1,2);
dresult(end,2) = dresult(end-1,2);
figure('Name','derive profil zone centrale');
plot(dresult(:,1),dresult(:,2));
[mini,ind] = min(dresult(:,2));
vi = dresult(ind,1);
hold on
scatter(vi,mini,'filled');
```

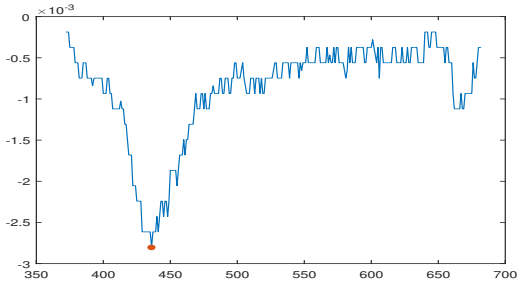


Figure 8: Position du point d'inflexion

Question 4

On trace la ligne noir de hauteur $v_h = 425$ dans l'image 9 il correspond à peu près à l'horizon de la route. Avec $\lambda = 500$ on obtint $k = 0.0440$. Par l'estimation de la distance de visibilité météorologique avec l'équation (5) on a $d_{visib00} = 68.1818m$.

```
figure('Name','horizon');
imad(420:425,:)=0;
imshow(imad)
vh = 425; lambda = 500;
k=2*(vi-vh)/lambda;
dvisible = 3.0/k
```

Figure 9: Hauteur de l'horizon ligne entre v = 420 et 425

Question 5

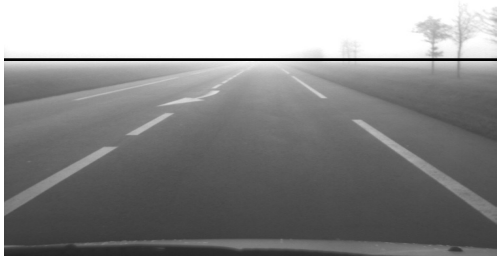
Pour estimer la distance de visibilité sur les deux autres images "F01.png" et "F02.png", on peut utiliser la structure des blocs de mon code et on remplace seulement le code dans la question 4 de la partie précédent "F00.png" en "F01.png" puis "F01.png" en "F02.png".
Il faut adapter la valeur de vh sinon on a des valeur négatifs. On calcule de façon heuristique vh comme étant la hauteur de l'horizon. Pour l'image "F01.png" on prend vh = 140 comme montré dans la figure 10(a), on obtient la distance de visibilité suivante $d_{visib01} = 37.5000m$
Quant l'image "F02.png" elle est en couleur. Afin d'assurer la généralité d'execution du code, on rectifie la question 4 de la partie précédente.

```
ima = imread('F02.png');
if size(ima,3) ==3
    imad = rgb2gray(double(ima)/255.0);
else
    imad = double(ima)/255.0;
end
```

On prend vh = 280 comme montré dans la figure 10(b). Pour des multitides des sélection de zones on obtient des distances de visibilité négatif et l'allure du profil loin du modèle théorique. On change vh à 260 on obtient avec la zone sélectionné en figure **??**la distance de visibilité $d_{visib02} = 93.7500$

Question 6

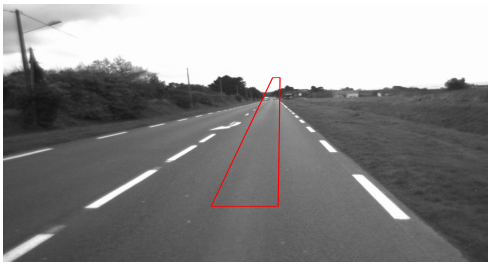
On peut détecter la présence de brouillard par la recherche de la position du point d'inflexion du profil verticale d'intensité. En fait d'aprés l'équation (4) : $k = 2\frac{v_i - v_h}{\lambda}$ on peut dire qu'il y a brouillard si la ligne de hauteur du point d'inflexion est sous la ligne d'horizon c'est à dire $v_i > v_h$ sinon il n'y a pas de brouillard dans l'image.



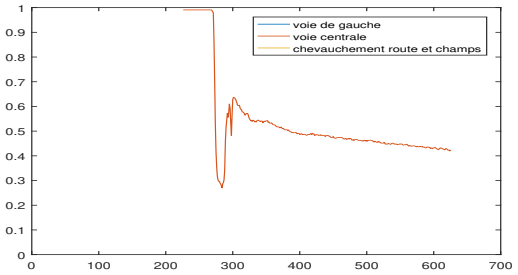
(a) image ”F01.png” ligne (140:145)

(b) image ”F02.png” ligne (140:145) (

Figure 10: Hauteur de l’horizon



(a) zone sélectionné



(b) profil associé

Figure 11: **profil** appliqué à l’image ”F02.png”

4 Restauration de la visibilité dans une image

Indépendamment de la détection du brouillard et de la mesure de la distance de visibilité, nous allons améliorer les contrastes atténués par le brouillard dans une image, en estimant le voile atmosphérique et en inversant la loi de Koschmieder.

Travail à effectuer

Question 1

En introduisant le voile défini par la variable $V = I_s(1 - e^{-kd})$ et en éliminant le produit de variables kd dans la loi de Koschmieder on obtient :

$$I = I_0(1 - \frac{V}{I_s}) + V$$

Question 2

On peut inverser le modèle précédent pour obtenir l’expression de l’image restaurée en fonction de l’image avec du brouillard, le voile et l’intensité du ciel.

$$I_0^{estimé} = I_s \frac{I - V}{I_s - V}$$

Question 3

En supposant que le ciel est la partie la plus lumineuse de l’image, on peut estimer l’intensité du ciel comme étant la moyenne des pixels de cette partie la plus lumineuse.

```
sorted_I = sort(imadgray(:));
n = 100;
Is = mean(sorted_I(end-n:end));
```

Question 4

On fait le lissage

```
dx = 5; dy = 66;
imad_filtred_med = medfilt2(imadgray,[dx dy],'symmetric');
V = .95*min(imadgray,imad_filtred_med);
```

Question 5

En utilisant l'équation du question 2 on arrive à restaurer la visibilité de l'image comme montré dans la figure 12

```
ima_est=Is*(imadgray - V)./(Is - V);
figure('Name','restauration');
imshow(ima_est);
```

Figure 12: image restauré

Question 6

On fait la correction gamma on mettant tous les valeurs de pixels à une puissance $\in [0, 1]$ par exemple 0.5 ce qui augmente leurs valeurs est devienne plus claires. On voit dans la figure 13 la restauration final de l'image.

```
ima_est = ima_est.^(.4);
figure('Name','correction gamma');
imshow(ima_est);
```

Question 7

Grâce à la structure des blocs de mon script, on remplace seulement le code dans la question 3 de cette partie "F00.png" en "F01.png" On obtient la figure 14

Question 8

On exécute les mêmes opérations que précédemment sur la nouvelle image tout en la convertir en niveau de gris.

(a) image avec brouillard

(b) image restauré et corrigé par transformation gamma

Figure 13: Restauration image ”F00.png”

(a) image avec brouillard

(b) image restauré et corrigé par transformation gamma

Figure 14: Restauration image ”F02.png”

```
ima = imread('F03.png');
imadgray = rgb2gray(double(ima)./255.0);
Is = imadgray(1,1,1);
dx = 5; dy = 66;
imad_filtred_med = medfilt2(imadgray,[dx dy],'symmetric');
V = .95*min(imadgray,imad_filtred_med);
ima_est=Is*(imadgray - V)./(Is - V);
ima_est = ima_est.^(.6);
figure('Name','correction gamma F03 image');
imshow(ima_est);
```

(a) image originale

(b) image restauré

Figure 15: Filtrage de **imadgray** avec deux filtres