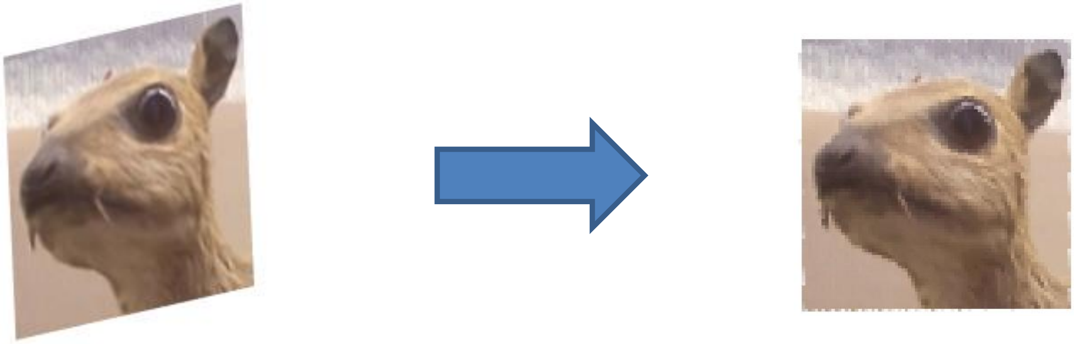
Informatique

7
Matrices de pixels et images

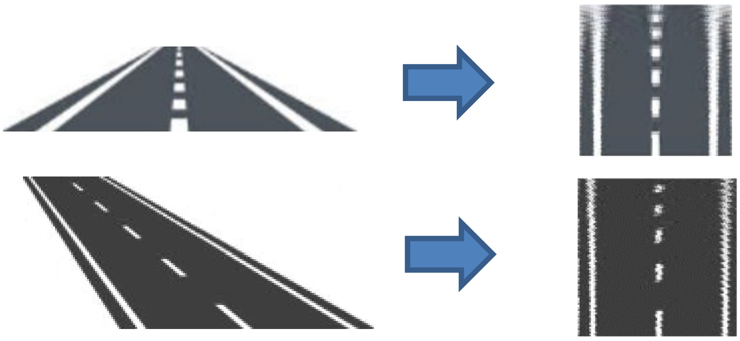
TD 7-16
Recadrage bilinéaire

Contexte

On souhaite créer un algorithme qui recadre un document scanné en définissant manuellement les 4 coins de l'image pour en déduire l'image recadrée. Nous allons mettre en place l'algorithme réalisant ce travail par interpolation bilinéaire :

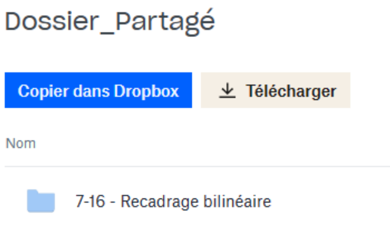


La méthode que nous allons programmer fonctionne bien s'il n'y a pas trop d'effets de perspective sur l'image (l'utilisation d'un scanner rend le résultat parfait). Sinon, la qualité devient moyenne et vous remarquerez sur les deux exemples ci-dessous qu'il n'y a pas de remise à l'échelle des lignes pointillés :



Affichage de l'image

Afin d'assurer un fonctionnement rapide sur tous les ordinateurs, je vous mets à disposition un dossier à télécharger COMPLETEMENT, soit le dossier contenant tous les fichiers, et non les fichiers pris séparément
Sans ouvrir le dossier, faite juste « Télécharger – Téléchargement direct » puis mettez ce dossier dans votre répertoire personnel.



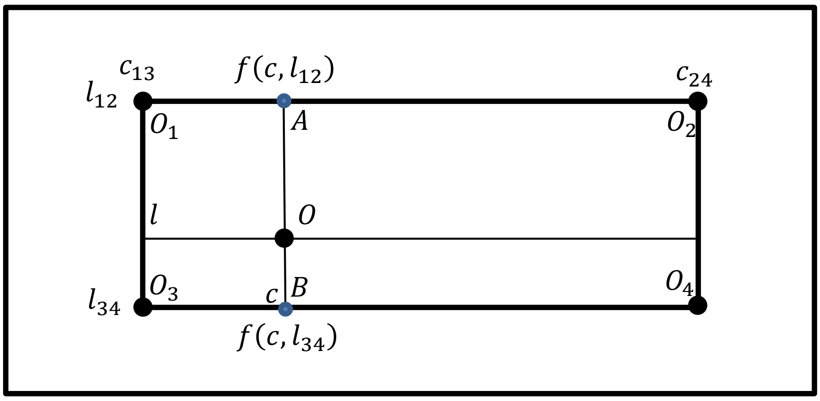
LIEN

Si le téléchargement est sous forme de Rar, Zip… Pensez à dézipper l'archive afin d'avoir le dossier voulu !
Vous y trouverez un code élève et l'image « Image.bmp » ainsi que sa version numpy « Image.npy ».

Question 1: Télécharger et exécuter le code fourni qui créera et affichera l'image « Image » sous Python

Interpolation bilinéaire

On considère 4 points dits « Origine » O_1, O_2, O_3 et O_4 dans une image aux lignes et colonnes (l_{ij}, c_{ij}) précisées ci-dessous :



Les 4 points peuvent être placés dans l'image (pas forcément sur les bords), mais :

- O_1 et O_2 doivent être à la même ligne, de même que O_3 et O_4
- O_1 et O_3 doivent être à la même colonne, de même que O_2 et O_4

A chaque point O_i de cette image est associée un n-uplet noté P_i. On souhaite déterminer le n-uplet $P = f(l, c)$ associé à un point quelconque $O(l, c)$ de l'image par interpolation bilinéaire. Nous avons réalisé cette démarche dans le cours sur l'agrandissement avec P_i les triplets RGB de points encadrants une zone à remplir, je ne rappelle donc ici que la fonction utile pour la suite (démonstration en annexe):

$$P = f(O) = f(l, c) = \frac{1}{(l_{34} - l_{12})(c_{24} - c_{13})} \begin{bmatrix} l_{34} - l & l - l_{12} \end{bmatrix} \begin{bmatrix} P_1 & P_2 \\ P_3 & P_4 \end{bmatrix} \begin{bmatrix} c_{24} - c \\ c - c_{13} \end{bmatrix}$$

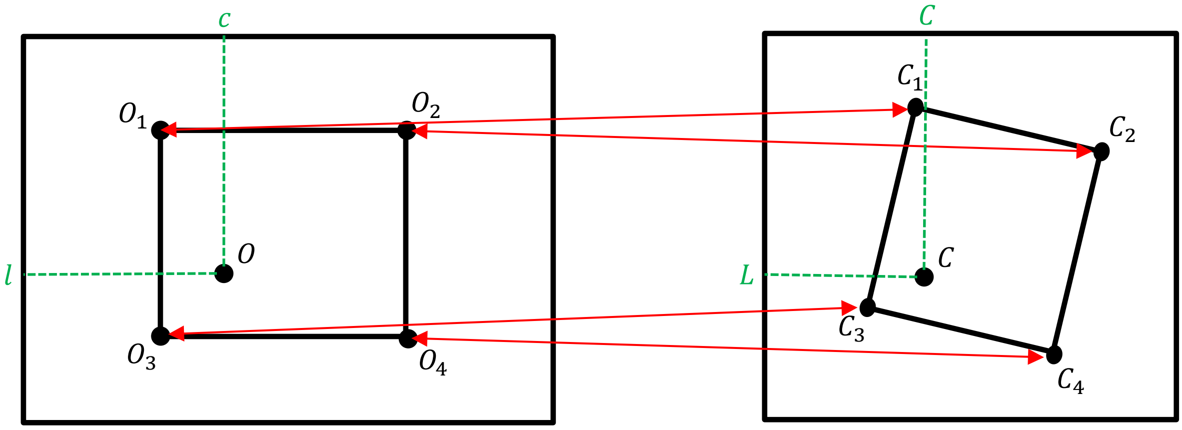
On obtiendra par exemple $f(O_i) = P_i$, et si $C = \frac{1}{4}\sum O_i$ est le centre des 4 points P_i, $f(C) = \frac{1}{4}\sum P_i$.

Recadrage

Soit une image blanche de dimensions (N_l, N_c) dite « origine » et l'image à recadrer dite « cible » de dimensions (N_L, N_C). En parcourant toute l'image « origine », on va pour chaque point $O(l, c)$ déterminer par interpolation bilinéaire les coordonnées du point de l'image « cible » à recadrer $C(L, C)$. On appliquera alors au point O de l'image origine la couleur du point C dans l'image cible si ce point en fait partie.

Autrement dit, en définissant $P_i = C_i$ les coordonnées des 4 coins dans l'image à recadrer (obtenues manuellement dans ce TD par observation avec la souris) :

- On calcule les coordonnées $L, C = f(l, c)$ du pixel associé dans l'image cible
- A partir d'une image d'origine blanche, on applique la couleur dans l'image origine $ImO[l, c]$ du pixel $ImC[L, C]$ si $L \in [0, N_L - 1]$ et $C \in [0, N_C - 1]$ (laissé blanc sinon)



Programmation

On ne travaillera qu'avec des listes sauf pour les valeurs des pixels qui seront des arrays.

On souhaite pouvoir réaliser le calcule $[a \quad b]\begin{bmatrix} P_1 & P_2 \\ P_3 & P_4 \end{bmatrix}$ donnant la liste $[A, B] = [aP_1 + bP_3, aP_2 + bP_4]$ avec P_i des n-uplets représentés sous forme de listes de taille n.

Question 2: Proposer une fonction Prod_MV(M,V) prenant en arguments une matrice M (liste de lignes (listes) de valeurs Pi (listes)) et un vecteur V (liste) et renvoyant le produit attendu

Vérifier :

```
>>> M = [[[1,2],[3,4]],[[5,6],[7,8]]]

>>> V = [1,2]

>>> Prod_MV(M,V)
[[7, 10], [19, 22]]
```

On souhaite alors réaliser le calcul $[A,B][a,b] = aA + bB$ avec A et B des n-uplets (issus du calcul précédent).

Question 3: Proposer une fonction Prod_LnV(Ln,V) prenant en arguments une liste de n-uplets Ln et un vecteur V (liste) et renvoyant le produit attendu

Vérifier :

```
>>> Ln = [[1,2],[3,4]]

>>> V = [1,2]

>>> Prod_VV(Ln,V)
[7, 10]
```

Dans la suite, on définit :

- LO=[O1,O2,O3,O4] : liste des 4 points de l'image origine
- LC=[C1,C2,C3,C4]=[P1,P2,P3,P4] : liste des 4 points de l'image cible

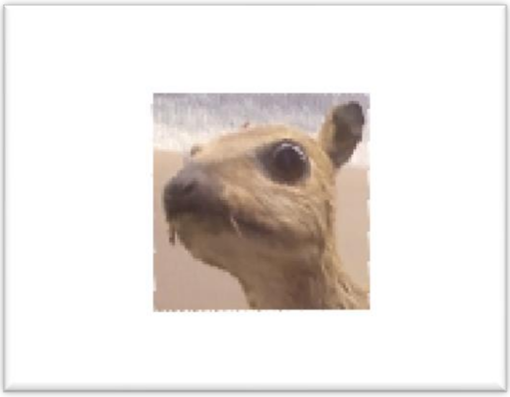
Question 4: Proposer une fonction LC_Bilin(l,c,LO,LC) prenant en argument les coordonnées l,c d'un point de l'image origine, les listes LO et LC, et renvoyant les coordonnées L,C associées au couple l,c par interpolation bilinéaire

Vérifier que la propriété $f(O_i) = P_i = C_i$ est vérifiée.

Question 5: Proposer la fonction Recadrage(im,LO,LC,Nl,Nc) réalisant le recadrage de l'image im sur une nouvelle image de dimensions Nl,Nc

On souhaite obtenir une image recadrée de dimensions 100x100 contenant une bande blanche de largeur de 10 pixels sur tout le tour.

Question 6: Mettre en place le code réalisant et affichant le recadrage attendu

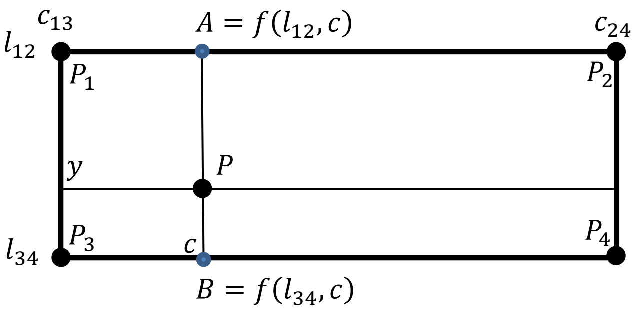


Question 7: Faire les application avec les images Route1 et Route2 en enlevant la zone blanche périphérique

Question 8: Créer im_resize(im,NL,NC) qui redimensionne im en une nouvelle image de dimensions NL,NC qui sera renvoyée, puis vérifier qu'elle fonctionne

Remarquez que vous avez maintenant à disposition un nouvel algorithme qui vous permet de réaliser les transformations de rotation, réduction et agrandissement en ayant la possibilité de changer les proportions de l'image.

Annexe

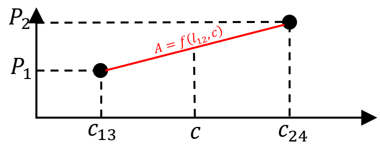


On définit la fonction $f(l, c)$ la fonction qui renvoie la valeur P à attribuer à un point de coordonnées (l, c). On a :

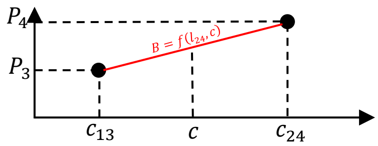
$$\begin{cases} f(l_{12}, c_{13}) = P_1 \\ f(l_{12}, c_{24}) = P_2 \\ f(l_{34}, c_{13}) = P_3 \\ f(l_{34}, c_{24}) = P_4 \end{cases}$$

Par interpolation linéaire, on a :

$$A = f(l_{12}, c) = \frac{c_{24} - c}{c_{24} - c_{13}} P_1 + \frac{c - c_{13}}{c_{24} - c_{13}} P_2$$

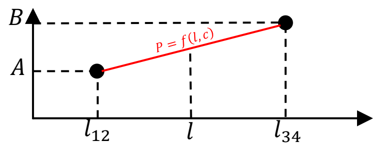


$$B = f(l_{24}, c) = \frac{c_{24} - c}{c_{24} - c_{13}} P_3 + \frac{c - c_{13}}{c_{24} - c_{13}} P_4$$



Soit, par interpolation entre A et B :

$$f(l, c) = \frac{l_{34} - l}{l_{34} - l_{12}} A + \frac{l - l_{12}}{l_{34} - l_{12}} B$$



En développant :

$$\begin{aligned} f(l, c) &= \frac{l_{34} - l}{l_{34} - l_{12}} \left(\frac{c_{24} - c}{c_{24} - c_{13}} P_1 + \frac{c - c_{13}}{c_{24} - c_{13}} P_2 \right) + \frac{l - l_{12}}{l_{34} - l_{12}} \left(\frac{c_{24} - c}{c_{24} - c_{13}} P_3 + \frac{c - c_{13}}{c_{24} - c_{13}} P_4 \right) \\ &= \frac{1}{(l_{34} - l_{12})(c_{24} - c_{13})} [(l_{34} - l)[(c_{24} - c)P_1 + (c - c_{13})P_2] \\ &\qquad + (l - l_{12})[(c_{24} - c)P_3 + (c - c_{13})P_4]] \\ &= \frac{1}{(l_{34} - l_{12})(c_{24} - c_{13})} \left[(l_{34} - l) \left[[(c_{24} - c) \quad (c - c_{13})] \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \right] \right. \\ &\qquad \left. + (l - l_{12})[(c_{24} - c) \quad (c - c_{13})] \begin{bmatrix} P_3 \\ P_4 \end{bmatrix} \right] \end{aligned}$$

$$\boxed{f(l, c) = \frac{1}{(l_{34} - l_{12})(c_{24} - c_{13})} [l_{34} - l \quad l - l_{12}] \begin{bmatrix} P_1 & P_2 \\ P_3 & P_4 \end{bmatrix} \begin{bmatrix} c_{24} - c \\ c - c_{13} \end{bmatrix}}$$