

Traitement de l'image pour la vision artificielle

Introduction à Scilab

Scilab est un logiciel de calcul axé sur la manipulation de matrices. Il est gratuit (développé par l'INRIA et l'ENPC) et disponible à l'adresse suivante : <http://www.scilab.org>.

Créez un répertoire de travail et lancez Scilab en tapant dans une console les lignes suivantes :

```
mkdir scilab    crée le répertoire (make directory)
cd scilab      se déplace dans le répertoire (change directory)
scilab         lance scilab
```

1 Manipulation de vecteurs et de matrices

Tapier dans Scilab les commandes indiquées après la flèche -->.

```
--> 5
--> (3.4+9/5)*3
--> abs(5)+cos(3)    valeur absolue, cosinus
--> %pi              %pi est la constante π
--> 5^2              puissance
--> ans + 1          ans = dernier résultat (answer)
--> sqrt(16)         racine carrée (square root)
--> %i^2             %i = i (complexe)
--> abs(1+%i)        module
--> x=5+3            stocke le résultat dans la variable x
--> x^2
--> y=3+4 ;          point-virgule : effectue le calcul sans afficher le résultat
--> y
--> y=5 ; y^2        plusieurs calculs sur la même ligne
--> y=1+2+3+4+5+6+...
--> 7+8+9+10         un calcul sur plusieurs lignes
--> v=[1,2,3]         vecteur ligne
--> w=[1 ; 2 ; 3]     vecteur colonne
--> v'                transposition
--> u=[4,5,6] ;      [v,u] concaténation de deux vecteurs
--> t=3 :7            début :fin
--> t=1 :2 :10        début :pas :fin
--> t=0 :0.1 :1
--> linspace(0,10,4) 4 nombres régulièrement espacés entre 0 et 10
--> u=t+1
--> u=2*t
--> u+t
--> u*t'              produit scalaire
--> A=[1,2 ; 3,4 ; 5,6] matrice
--> B=[1,2,3 ; ...
--> 4,5,6 ; ...
--> 7,8,9]
--> size(A)           =[nb de lignes, nb de colonnes]
--> ones(3,4)
--> ones(A)           matrice de même dimension que A
--> zeros(3,4)
--> eye(5,5)          matrice identité (I prononcé à l'anglaise)
--> rand(3,4)         matrice aléatoire (uniforme entre 0 et 1)
--> A*[1,2,3 ; 4,5,6] multiplication matricielle
--> A+B              dimensions non compatibles!
```

```

--> B(3,2)          élément à la ligne 3, colonne 2
--> B( :,2)         deuxième colonne
--> B(2, :)         deuxième ligne
--> B(2 :3,1 :2)    matrice extraite : intersection des lignes 2 à 3 avec les colonnes 1 à 2
--> B(3,2)=10 ;     change la valeur de l'élément ligne 3 colonne 2
--> B
--> B( :,2)=[-1 ; -3 ; -5] ;
--> B

```

Exercice

Construire les matrices suivantes :

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 7 \end{pmatrix}, B = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}, C = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 2 & 0 & 1 & 0 & 0 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 & 0 & 1 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

```

--> w=1 :2 :9
--> w($)          dernier élément
--> w($-1)       avant-dernier élément
--> int(5.2)     partie entière
--> x=[1,2,3] ;x.*x  multiplication vectorielle terme à terme

```

2 Graphiques

```

--> x=0.1 :0.1 :10 ;y=sin(x)./x ;
--> plot2d(x,y)          graphique
--> z=cos(x)./x ;
--> clf() ;              efface le graphique (clear figure)
--> plot2d(x,y) ;plot2d(x,z)  affiche 2 graphiques superposés
--> clf() ;plot2d([x ;x] ', [y ;z] ')  idem
--> clf() ;plot2d(x,z,style=-5)
--> clf() ; --> plot2d(x,z,rect=[0,0,4,4])  rect=[xmin,ymin,xmax,ymax]

```

3 Scripts

Un script est un fichier contenant une liste de commandes Scilab. Pour créer un nouveau script, cliquez sur « Editor » dans le menu. Tapez le script suivant dans l'éditeur :

```

t=linspace(0,1,10) ;
x=sin(t.^2) ;
plot2d(t,x) ;

```

Enregistrez sous le nom « script1.sce » (menu File > Save as). Revenez dans Scilab et exécutez le script ainsi :

```

--> exec("script1.sce")

```

Exercice

Modifier le script pour afficher 20 points au lieu de 10

4 Programmation

Écrivez les programmes suivants dans des scripts et exécutez-les.

Conditions

```

x=rand()*1.1;
if x<0.5 then p="pile";      si ... alors
elseif x<1 then p="face";  sinon si ...alors
else p="tranche";          sinon
end                          fin du si
p
  Boucle while
x=rand();
n=0;
while x<0.9 do  tant que ...faire
n=n+1;
x=rand();
end              fin de la boucle
  Boucle for
n=1 :10;
m=1;
for i=n  pour i prenant chaque valeur du vecteur n
m=m*i;
end      fin de la boucle

```

5 Fonctions

Les fonctions sont définies dans des scripts. Voici par exemple la fonction factorielle :

```

function f=factorielle(n)  résultat = nom_de_la_fonction(arguments)
f=prod(1 :n);
endfunction                fin de la fonction

```

Enregistrez le fichier sous le nom `factorielle.sci` (les fichiers de scripts ont généralement l'extension `.sce`, les fichiers de fonctions ont l'extension `.sci`), puis exécutez-le (`exec('factorielle.sci')`). Vous pouvez maintenant utiliser la nouvelle fonction `factorielle` :

```

--> factorielle(4)
--> factorielle(8)

```

Si l'on modifie la fonction dans le fichier, il faut enregistrer et recharger le fichier dans Scilab (`exec`) avant de pouvoir utiliser la fonction (sinon on utilise l'ancienne version).

Une fonction peut renvoyer plusieurs résultats et prendre plusieurs arguments (scalaires, vectoriels ou matriciels) :

```

function [x,y]=polaire(theta,r)
x=r*cos(theta);
y=r*sin(theta);
endfunction

```

Une fonction peut s'appeler elle-même (elle est dite récursive) :

```

function f=factorielle2(n)
if n<=1 then
f=1;
else
f=n*factorielle2(n-1)
end
endfunction

```

5.0.1 Histogramme

L'histogramme d'une image représente les fréquences relatives des différentes valeurs des niveaux de gris d'une image. Formellement, l'histogramme h d'une image I est défini par :

$$h(u) = n_u$$

avec n_u le nombre de pixels de l'image I ayant pour niveau de gris u .

- Développer une fonction permettant de calculer l'histogramme d'une image de taille $[N, M]$,
- Charger l'image "couloir" en tapant l'instruction (en ayant activé la boîte à outils SIP) :

```
ImCouloir=imread('couloir.jpg');
```

ou

```
[ImCouloir, CarteCoul]=imread('couloir.jpg');
```

pour avoir la carte des couleurs associée.

L'image couloir.jpg est alors stockée en mémoire dans la variable ImCouloir.

- **Regarder la démonstration de SIP avec la commande** `exec(SIPDEMO)`
- **Visualiser l'image** ImCouloir.
- **Calculer l'histogramme de** ImCouloir .
- **Visualiser l'histogramme. Quelles informations déduisez-vous de la visualisation de cette histogramme.**
- **Visualiser l'histogramme en utilisant la fonction de la toolbox SIP de Scilab :**

```
histplot(nclasses, ImCouloir);
```

affiche l'histogramme de ImCouloir avec nclasses le nombre de classes également espacées.