Dimitri Charneux
Rémi Kroll

Rendu tp: moments d'une forme

Code R

Dans le fichier « rdfMoment.R », il y a des fonctions permettant de calculer un moment d'une forme.

rdfSurface va effectuer la somme des valeurs de la matrice passée en paramètres, c'est-à-dire tout les 1. Cela va nous donner la surface de la matrice donnée en paramètre.

rdfMoment va créer le moment d'ordre p q. La somme est décomposée de cette manière: on créé un vecteur x de taille 1 à la taille max des x. On multiplie par p chaque x du vecteur. On fait la même chose avec les y. Ensuite avec les deux vecteurs x et y, on créé deux matrices en combinant les vecteurs lignes et les vecteurs colonnes. Puis on les multiplies ensemble avec la matrice im passée en paramètres, ce qui nous donne l'indice du moment à l'ordre (p,q).

rdfMomentCentre calcule le moment centre à l'indice (p,q) de la matrice im : la formule est la même que rdfMoment sauf qu'on enlève cx et cy pour chaque vecteur ce qui correspond au barycentre en x et le barycentre en y divisé par la surface de im. Cela permet de calculer un indice indépendamment de la position de l'objet.

L'intérêt de ce calcul est de faire une multiplication de somme en une seule ligne. On veut minimiser la complexité du calcul.

Axes principaux d'inertie

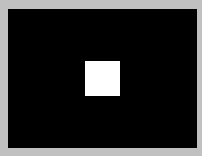
La matrice d'inertie se calcule à l'aide du tenseur d'inertie. Ici, nous calculons le tenseur d'inertie à deux dimensions. On créé une matrice à deux dimensions avec comme valeurs les trois indices des moments centrées d'ordres 2, c'est-à-dire, [u20,u11,u11,u02]. Ensuite on calcule les valeurs propres avec la fonction eigen. La valeur la plus faible correspond a l'axe de l'objet qui tourne le mieux et inversement la valeur la plus forte correspond à l'axe où l'objet tournera le plus difficilement. On observe que les valeurs propres des vecteurs restent les même et que par contre, les vecteurs propres sont dépendant de l'orientation des rectangles, ils ont tous des valeurs différentes.

Entre les deux rectangles diagonales, il y a des différences de couleurs de pixels au niveau des bords du rectangle. Le rectangle D lissé contient plus de pixels mi blanc mi noir pour rendre le trait plus lisse, c'est le phénomène d'anticrénelage par rapport à l'autre rectangle diagonal non lissé possédant le phénomène contraire de crénelage.

On remarque que pour le rectangle diagonal lissé, il y a une augmentation de 10% sur les valeurs propres par rapport au rectangle diagonal non lissé. C'est à cause de l'anticrénelage.

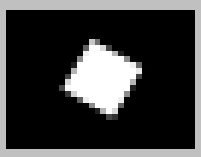
Carré côté 6 :

```
$values
[1] 105 105

$vectors
     [,1] [,2]
[1,]    0   -1
[2,]    1    0
```



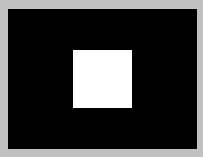
Rotation 30deg :

```
$values
[1] 843.2815 842.4202

$vectors
          [,1]       [,2]
[1,] 0.2975906 -0.9546936
[2,] 0.9546936  0.2975906
```



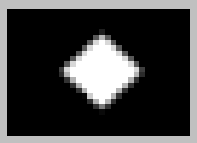
Carré côté 10 :

```
$values
[1] 825 825

$vectors
     [,1] [,2]
[1,]    0   -1
[2,]    1    0
```



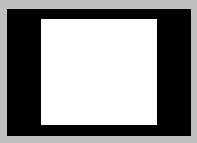
Rotation 45deg :

```
$values
[1] 841.5171 838.5359

$vectors
           [,1]       [,2]
[1,] -0.7771076  0.6293678
[2,] -0.6293678 -0.7771076
```



Carré côté 20 :

```
$values
[1] 13300 13300

$vectors
     [,1] [,2]
[1,]    0   -1
[2,]    1    0
```



On observe que les deux valeurs propres des différents carrés sont toujours identiques entre eux. Lorsque le vecteur est orienté , la valeur du vecteur est réel sinon la valeur du vecteur est entière. Donc un attribut de forme pour le carré pourrait être l'égalité des valeurs propres de la forme.

Moments normalisés

Nous avons vu que les valeurs propres des carrées sont identiques entre elles mais en fonction de la taille du carré, elles diffèrent : les valeurs propres d'un carré de coté 6 sont plus petites que celles du carré du coté 20. La normalisation des moments va permettre de rendre les

valeurs propres indépendant de l’échelle de l'objet.

```
Carre 6          :  0.08101852  /  0.08101852
Carre 10         :  0.0825  /  0.0825
Carre 20         :  0.083125  /  0.083125
```

On observe, pour les touts les carrés de taille différentes, que les valeurs sont quasiment identique.

Valeurs propres normalisées calculées à partir des moments principaux d'inertie normalisés :

Rectangle vertical :

```
$values
[1] 0.33203125 0.01953125
```

Rectangle horizontal :

```
$values
[1] 0.33203125 0.01953125
```

Rectangle diagonal :

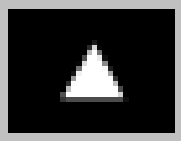
```
$values
[1] 0.38585018 0.01753864
```

Rectangle diagonal lisse :

```
$values
[1] 0.33503454 0.02395988
```

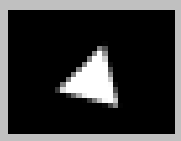
axeInertieNormalise de triangle coté 10 :

```
$values
[1] 0.10077660 0.09484009

$vectors
           [,1]       [,2]
[1,] -0.2038465 -0.9790029
[2,]  0.9790029 -0.2038465
```



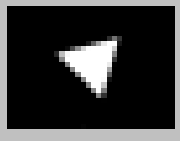
axeInertieNormalise Triangle Rotation 15deg :

```
$values
[1] 0.10033918 0.09483242

$vectors
           [,1]       [,2]
[1,] -0.4667712 -0.8843781
[2,]  0.8843781 -0.4667712
```



axeInertieNormalise Triangle Rotation 45deg :

```
$values
[1] 0.10050438 0.09515709

$vectors
           [,1]       [,2]
[1,] -0.8082025 -0.5889047
[2,]  0.5889047 -0.8082025
```



axeInertieNormalise Triangle Rotation 60deg :

```
$values
[1] 0.10154882 0.09336958
```

```
Carre 10          :  0.0825  /  0.0825
Carre 10, 30deg   :  0.08411031  /  0.08402441
Carre 10, 45deg   :  0.08543340  /  0.08513074
```

On remarque qu'en diagonalisant la matrice d'inertie à partir des moments centrés normalisés, les valeurs propres d'une forme deviennent les même indépendant de leur rotation. On a quasiment les même valeurs propres à 0,01 près. On peut en conclure que ces moments peuvent être utilisés comme attribut de forme. On pourra donner un attribut pour le triangle de côté 10 équilatéral. On retrouvera le même attribut pour chaque triangle équilatéral qui possède où non une rotation ou un changement d'échelle par rapport à celui de base de taille 10. On aura aussi un attribut pour un rectangle et pour un carré,qui suivront le même principe que le triangle équilatéral.

Moments invariants

Comparaison entre le numéro 1 et 7 (en gras les ressemblances) :

```
Calculs de moments invariants pour le 1
1 : 0.6165951
2 : 0.2460521
3 : 0.01457684
4 : 0.002890858
5 : 1.288171e-05
```

```
Calculs de moments invariants pour le 7
1 : 0.5951606
2 : 0.1605388
3 : 0.1078302
4 : 0.02111472
5 : 0.0005330942
```

Comparaison entre le numéro 6 et 9 (en gras les ressemblances) :

```
Calculs de moments invariants pour le 6
1 : 0.3838055
2 : 0.01669642
3 : 0.001734057
4 : 0.000320043
5 : -1.923754e-07
```

```
Calculs de moments invariants pour le 9
1 : 0.3879763
2 : 0.01808315
3 : 0.002671323
4 : 0.0005338399
5 : -5.958311e-07
```

```
Calculs de moments invariants pour le 8
1 : 0.376284
2 : 0.01920328
3 : 0.000136424
4 : 1.487603e-05
5 : -6.69642e-10
```

```
Calculs de moments invariants pour le 3
1 : 0.5273825
```

```
2 : 0.08907071
3 : 0.016224
4 : 0.004589431
5 : -3.616123e-05
```

```
Calculs de moments invariants pour le 0
 1 : 0.4561521
 2 : 0.01753315
 3 : 7.632676e-08
 4 : 1.623057e-07
 5 : -1.747216e-14
```

Au vu des résultats, après le calcul des 5 premiers moments invariants de Hu pour les 10 chiffres de 0 à 9, on remarque que les 5 premiers moments invariants sont, pour la plupart, tous différents. Certains moments invariants se ressemblent comme vous pouvez le voir ci dessus avec la comparaison du chiffre 6 et 9, et 1 et 7. Mais pour ces exemples, il y a toujours un des 5 premiers moments invariants assez différent pour qu'on ne puisse pas les confondre.

On en conclut qu'il est possible de différencier ces images de chiffres avec les moments invariants même si, sur certains cas, il y a des ressemblances entre les moments invariants.