Compte-rendu : Reconnaissance de formes.
Jérôme Dhersin – Yassine Badache
TP1 – Moments d'une forme.

Code R.

Le calcul des double sommes nécessaires à l'estimation des moments géométriques est calculée par multiplication de matrices. Le programme prend trois matrices, qu'il multiplie matriciellement entre elles : les matrices correspondant aux X, aux Y et à l'image. L'avantage de cette technique est qu'elle permet de décomposer le calcul et le rendre plus efficace.

On multiplie, dans l'ordre, la matrice X à la matrice image, puis ce résultat à la matrice Y pour obtenir la double somme des moments géométriques d'une forme.

Axes principaux d'inertie.

La fonction *rdfMatriceInertie* calcule les moments $M_{11,}$ M_{20} et M_{02} pour simplement les stocker dans une matrice, au format adapté :

M_{20} M_{11}
M_{11} M_{02}

Les moments principaux et axes d'inertie obtenus sont les suivants :
Pour rappel, « les valeurs propres du tenseur d'inertie représentent les moments principaux d'inertie, tandis que ses vecteurs propres définissent les axes principaux d'inertie ».

Rectangle H :

Axes principaux d'inertie : [1, 0] et [0, 1]. (Axe des abscisses et ordonnées)
Moments principaux d'inertie : 1360, 80.

Rectangle V :

Axes principaux d'inertie : [0, 1] et [1, 0]. (Axe des abscisses et ordonnées)
Moments principaux d'inertie : 1360, 80.

Rectangle D non-lisse :

Axes principaux d'inertie : [0.7071068, 0.7071068], et [-0.7071068, 0.7071068].
Moments principaux d'inertie : 1298, 59.

Rectangle D lisse :

Axes principaux d'inertie : [0.7061774, -0.7080350] et [-0.7080350, -0.7061774].
Moments principaux d'inertie : 1393.74269, 99.67303.

La différence entre les deux rectangles est que le rectangle non-lisse est une *image binaire* : ses pixels sont tous uniformes, contrairement au rectangle lisse. Sa masse est donc homogène en tout point du rectangle, ce qui n'est pas le cas du rectangle lissé.

La conséquence directe en est que le calcul des moments géométriques du rectangle en devient totalement différente, le poids de chaque pixel pouvant adopter des valeurs inférieures à 1.

Moments d'inertie des carrés :

Carré de côté 6 : 105, 105.
Carré de côté 10 : 825, 825.
Carré 30° rotation : 843.2815, 842.4202.
Carré 45° rotation : 841.5171, 838.5359.
Carré de côté 20 : 13300, 13300.

On remarque que si le carré n'a subi aucune rotation, les moments principaux d'inertie sont les mêmes. De plus, en cas de rotation, on peut observer une différence très légère. La taille du carré a une incidence sur le moment en lui-même.
Les moments d'inertie se trouvent être un attribut de forme efficace pour repérer un carré : s'ils sont égaux, ça en est un, autrement il y a de fortes chances que ça en soit un si les valeurs sont très proches.

/!\ : Les carrés ayant subis des rotations ne sont PAS uniformes, c'est-à-dire que, comme le rectangle lissé, ils ne sont pas de type binaire, ce qui peut potentiellement amener les erreurs légères entre les deux mentionnées plus haut. D'un autre côté, ne pas lisser un carré lors d'une rotation revient à perdre de l'information.

Moments normalisés.

La fonction *rdfMomentCentreNormalise*, comme son nom l'indique, calcule les moments centrés et les normalise d'après la formule vue en cours.

Moments normalisés d'ordre 2 :

Carré de côté 6 : 0.08101852.
Carré de côté 10 : 0.0825.
Carré de côté 20 : 0.083125.
Carré 30° rotation : 0.003928728.
Carré 45° rotation : 0.003086341.
Carré horizontal : 0.006484985.
Carré vertical : 0.006484985.
Rectangle diagonal : 0.06342831.
Rectangle diagonal lisse : 0.04725043.

Moments principaux d'inertie via moments normalisés :

Rectangle vertical : 0.3320312, 0.01953125.
Rectangle horizontal : 0.3320312, 0.01953125.
Rectangle diagonal: 0.2016944, 0.2016944.
Rectangle diagonal lisse : 0.1790886, 0.1799058.
Carré de taille 10 : 0.0825, 0.0825.
Carré de taille 20 : 0.083125, 0.083125.
Carré 30° rotation : 8.403202e-02, 8.410270e-02.
Carré 45° rotation : 0.0853135108, 0.0852506202.

Triangle de côté 10 : 0.095086775, 0.100529916.
Triangle 15° rotation : 0.095086775, 0.100529916.
Triangle 45° rotation : 0.098649894, 0.097011576.
Triangle 60° rotation : 0.093866912, 0.101051482.
Triangle de côté 20 : 0.096299491, 0.094765998.

On remarque que pour un objet dont les côtés sont égaux (par exemple, les carrés ou les triangles équilatéraux), les moments principaux d'inertie sont sensiblement égaux, indépendamment de la rotation effectuée par l'objet.
On peut donc en conclure que les moments normalisés peuvent être utilisés comme attributs de forme pour détecter un objet, cet attribut étant invariant de la taille ou de la rotation de l'objet.

Moments invariants de Hu.

La fonction *rdfMomentsInvariants* permet de calculer les moments invariants de Hu.

Chiffre 0	Chiffre 1	Chiffre 2
0.4561521	0.6165951	0.5967148
0.01753315	0.2402702	0.132895
7.632676e-08	0.01457684	0.007449113
1.623057e-07	0.002890858	0.002414322
-4.173725e-14	1.632243e-05	-3.39682e-06

Chiffre 3	Chiffre 4	Chiffre 5
0.5273825	0.3466112	0.5135136
0.08906051	0.01658829	0.07346167
0.016224	0.01509043	0.005204035
0.004589431	0.0002348969	0.001957659
-3.596447e-05	6.145031e-07	

Chiffre 6	Chiffre 7	Chiffre 8
0.3838055	0.5951606	0.376284
0.01646183	0.1568431	0.01920328
0.001734057	0.1078302	0.000136424
0.000320043	0.02111472	1.487603e-05
-2.065263e-07	0.0005099411	-6.72692e-10

Chiffre 9
0.3879763
0.0176973
0.002671323
0.0005338399
-9.644495e-07

On remarque des similitudes entre les chiffres 6 et 9 au niveau de leurs moments invariants, comme les chiffres 1 et 7. Ces chiffres présentent des similitudes en terme de forme, indéniablement : le 6 n'est qu'une rotation du 9 et le 7 'contient' le 1 avec une légère rotation.
En conclusion, la première invariance de Hu à elle seule suffit à reconnaître des formes standard.

Conclusion.

Ces travaux pratique nous ont fait comprendre l'utilisation concrète des moments d'une forme pour la reconnaître, et utiliser les différents ordres d'un moment (moment d'ordre 2) et types (invariants, normalisés, de Hu) pour reconnaître une forme malgré sa rotation ou sa taille différant d'une forme standard.