

# TP2 : Codage d'un contour

Benjamin VAN RYSEGHEM  
François LEPAN

4 février 2013

## 1 Code Scilab

### 1.1 A quoi correspond l'argument bins de la fonction `rdfCalculeHistogramme1D` ?

```
????????????????????  
????????????????????  
????????????????????
```

### 1.2 Modifier la valeur de la variable seuil et commenter les résultats

La Fig. 1 correspond au changement de la valeur de seuil. De gauche à droite on retrouve l'image initiale, l'image binarisée avec un seuil de 0.35, l'image binarisée avec un seuil de 0.5 et enfin l'image binarisée avec un seuil de 0.7. Afin de mieux comprendre les changements d'affichage il faut se référer à l'histogramme (cf : Fig. 2) de l'image initiale.

En effet on observe 2 "pics" sur la courbe. Ceux-ci correspondent aux couleurs dominantes de l'image. Sachant que le noir correspond à la valeur 0 et le blanc à 1, on en déduit que le premier pic, situé entre 0.25 et 0.45, correspond à la couleur de fond gris foncé et le deuxième, situé entre 0.6 et 0.75, correspond à la couleur des cercles gris clairs.

Donc si on met le seuil de binarisation à 0.35 on lui dit que toutes les valeurs supérieures deviennent blanches. C'est pour cela que l'on peut voir plein de taches blanches sur le fond. Il en est de même pour un seuil à 0.7, toutes les valeurs inférieures à 0.7 deviennent noires c'est donc pour cela que l'on peut observer des taches noires à l'intérieur des cercles.

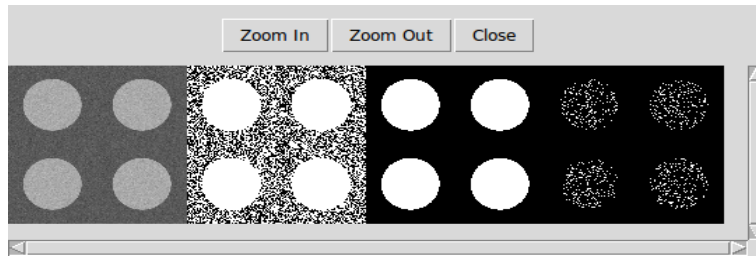


FIGURE 1 – De gauche a droite : Image initiale, Image binarisée avec un seuil de 0.35, Image binarisée avec un seuil de 0.5, Image binarisée avec un seuil de 0.5

### 1.3 Changer le signe des deux derniers arguments de la fonction `rdf-ClassifieurLineaire1D` et expliquer le résultat

En changeant le signe des deux derniers arguments on obtient la Fig. 3.

```
if a * i + b > 0 then
    result (y, x) = 1;
else
    result (y, x) = 0;
end
end
```

En observant la fonction on voit bien que au lieu que pour toutes les valeurs supérieur au seuil on met 1 on va a la place mettre 0.

Prenons un exemple : Si on prend  $a = 1$ ,  $b = -0.5$ .

Si on prend un  $i >$  au seuil 0.5  $\rightarrow i = 0.6$  on aura  $1 * 0.6 - 0.5 = 0.1 > 0$  et donc on mettra 1.

Si on prend un  $i <$  au seuil 0.5  $\rightarrow i = 0.4$  on aura  $1 * 0.4 - 0.5 = -0.1 < 0$  et donc on mettra 0

Par contre si on change de signe :  $a = -1$ ,  $b = 0.5$ .

Si on prend un  $i >$  au seuil 0.5  $\rightarrow i = 0.6$  on aura  $-1 * 0.6 + 0.5 = -0.1 < 0$  et donc on mettra 0.

Si on prend un  $i <$  au seuil 0.5  $\rightarrow i = 0.4$  on aura  $-1 * 0.4 + 0.5 = 0.1 > 0$  et donc on mettra 1.

C'est pour cela que l'affichage est inversé par rapport à la 3 ème image de la Fig. ?? .

### 1.4 À quoi correspond l'opération `[image, binaire]`

Cette opération correspond à la concaténation de image avec binaire. La condition pour que ces matrices puissent être concaténé est qu'elle fasse le même nombre de ligne.

Si on voulait les concaténer "de haut en bas" il suffirait de faire : `imshow ([binaire;binaire1]);` et la condition pour que cela marche est que les deux matrice possède le même nombre de colonnes.

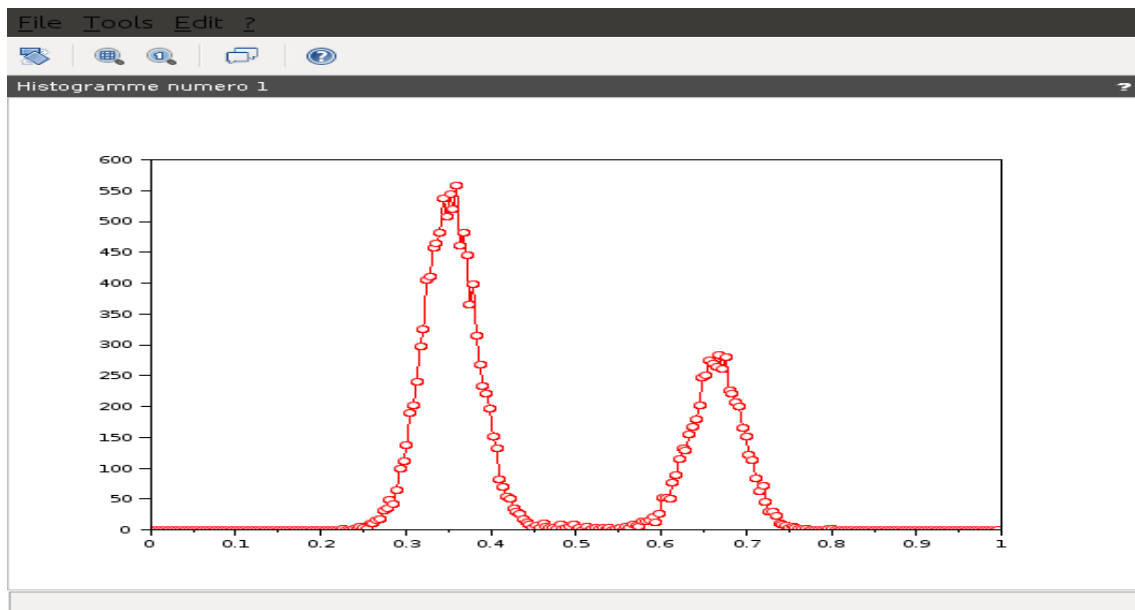


FIGURE 2 – Histogramme de l'image de base (cf : Fig. 1)

## 2 Histogramme des niveaux de gris

### 2.1 Calculer et afficher les histogrammes des niveaux de gris des images texture 0 à 4

Voici les 5 histogrammes correspondant aux figures :

- rdf-2-classes-texture-0.png Fig. 9
- rdf-2-classes-texture-1.png Fig. 10
- rdf-2-classes-texture-2.png Fig. 11
- rdf-2-classes-texture-3.png Fig. 12
- rdf-2-classes-texture-4.png Fig. 13

### 2.2 Pour chacune des ces images, déterminer le seuil qui permet de séparer au mieux les objets ronds du fond

En se basant sur les histogrammes précédemment calculé ainsi que la fonction *rdfClassifieurLineaire1D* nous avons pus déterminer les seuils qui permettent de retrouver Fig. 4 :

```
nom = "rdf-2-classes-texture-0.png";
image = im2double (imread (nom));

nom = "rdf-2-classes-texture-1.png";
image1 = im2double (imread (nom));

nom = "rdf-2-classes-texture-2.png";
```

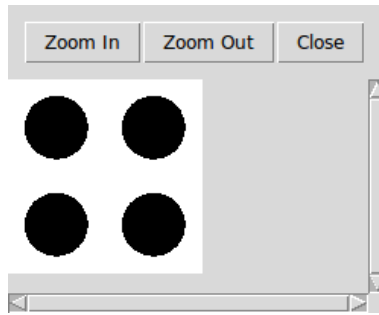


FIGURE 3 – Image résultant de l'inversion des deux derniers arguments de la fonction `rdfClassifieurLineaire1D` sur l'image `rdf-2-classes-texture-0.png`

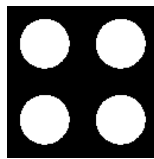


FIGURE 4 – Image référence

```
image2 = im2double (imread (nom));

nom = "rdf-2-classes-texture-3.png";
image3 = im2double (imread (nom));

nom = "rdf-2-classes-texture-4.png";
image4 = im2double (imread (nom));

seuil = 0.5;
binaire = rdfClassifieurLineaire1D (image, 1, -seuil);

seuil = 0.58;
binaire1 = rdfClassifieurLineaire1D (image1, 1, -seuil);

seuil = 0.31;
binaire2 = rdfClassifieurLineaire1D (image2, -1, seuil);

seuil = 0.39;
binaire3 = rdfClassifieurLineaire1D (image3, -1, seuil);

seuil = 0.45;
binaire4 = rdfClassifieurLineaire1D (image4, 1, -seuil);

imshow ([binaire,binaire1,binaire2,binaire3,binaire4]);
```

Après l'exécution du code suivant on obtient la Fig. 5

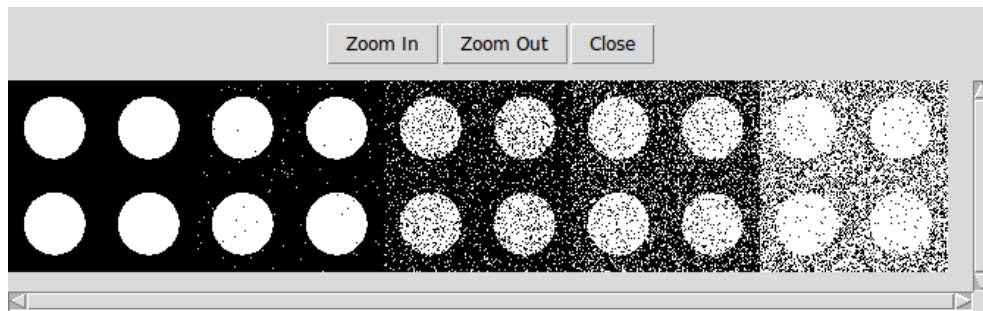


FIGURE 5 – Images texture-0 à 4 binarisées respectivement aux seuil : 0.5, 0.58, 0.31, 0.39, 0.45

### 2.3 Calculer la valeur absolue de la différence entre l'image binarisée et l'image de référence

```
resultat = abs(binaire-imgReference);
resultat1 = abs(binaire1-imgReference);
resultat2 = abs(binaire2-imgReference);
resultat3 = abs(binaire3-imgReference);
resultat4 = abs(binaire4-imgReference);

imshow ([imgReference,binaire,resultat;
         imgReference,binaire1,resultat1;
         imgReference,binaire2,resultat2;
         imgReference,binaire3,resultat3;
         imgReference,binaire4,resultat4]);
```

Les variables binaire, binaire1, binaire2, binaire3 et binaire4 correspondent aux images binarisées de la Fig. 5.

L'exécution du code suivant donne la Fig. 6 sur laquelle on peut observer de gauche à droite l'image de référence, l'image binarisée (respectivement de 0 à 4 de haut en bas) et les différents résultats (abs (image binarisée - image de référence)).

À la vue de ces résultats on peut dire que sur chaque résultat la couleur blanche est le "bruit" qui sépare l'image binarisée de la classification parfaite de ses éléments.

Pour les deux premières images il n'y a pas beaucoup de "bruit" et donc on distingue bien les cercles. Par contre pour le reste des images cela devient de moins en moins évident de distinguer ces cercles. On ne peut donc pas utiliser comme seul attribut de chaque pixel son niveau de gris pour la classification. Il faut plus d'information afin de mieux binariser chaque image et obtenir de meilleur résultat.

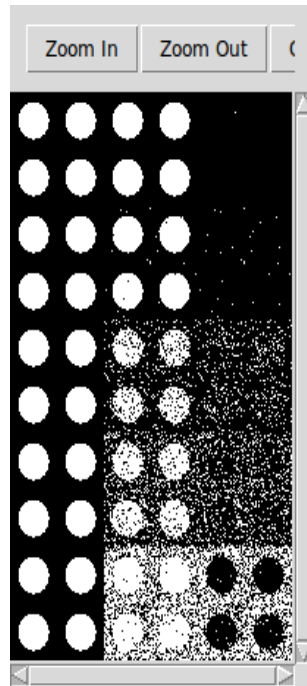


FIGURE 6 –

### 3 Histogramme des niveaux de texture

#### 3.1 Comment la fonction *rdfTextureEcartType* détermine le niveau de texture pour chaque pixel de l'image ?

```

????????????????????
????????????????????
????????????????????

```

#### 3.2 Calcule de l'histogramme de chacune de ces images de niveau de texture en utilisant un voisinage carré de taille 5x5

Voici les 5 histogrammes correspondant aux figures :

- rdf-2-classes-texture-0.png Fig. 14
- rdf-2-classes-texture-1.png Fig. 15
- rdf-2-classes-texture-2.png Fig. 16
- rdf-2-classes-texture-3.png Fig. 17
- rdf-2-classes-texture-4.png Fig. 18

// On charge les images comme pour la partie 2.2

```

imageNT = rdfTextureEcartType(image,5);
imageNT1 = rdfTextureEcartType(image1,5);

```

```

imageNT2 = rdfTextureEcartType(image2,5);
imageNT3 = rdfTextureEcartType(image3,5);
imageNT4 = rdfTextureEcartType(image4,5);

seuil = 0.7;
binaire = rdfClassifieurLineaire1D (imageNT, 1, -seuil);

seuil = 0.56;
binaire1 = rdfClassifieurLineaire1D (imageNT1, -1, seuil);

seuil = 0.57;
binaire2 = rdfClassifieurLineaire1D (imageNT2, -1, seuil);

seuil = 0.6;
binaire3 = rdfClassifieurLineaire1D (imageNT3, -1, seuil);

seuil = 0.55;
binaire4 = rdfClassifieurLineaire1D (imageNT4, -1, seuil);

imshow ([binaire,binaire1,binaire2,binaire3,binaire4]);

```

Après l'exécution du code suivant on obtient la Fig. 7

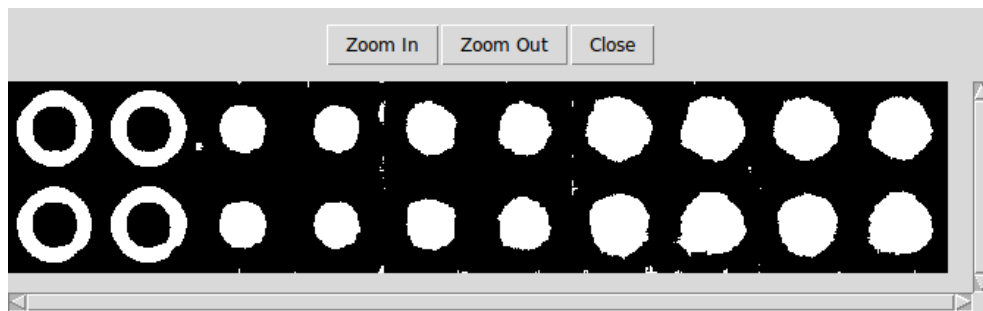


FIGURE 7 – Images texture-0 à 4 en niveau de texture puis binarisées respectivement aux seuil : 0.7, 0.56, 0.57, 0.6, 0.55

Ensuite afin de se rendre compte de la différence entre l'image binarisé et l'image de référence nous allons effectuer le même calcul et affichage des résultats que vu précédemment dans la partie 2.3.

Ce qui nous donnera la Fig. 8

Au vu de ces résultats nous pouvons????????????????????  
 ???  
 ???

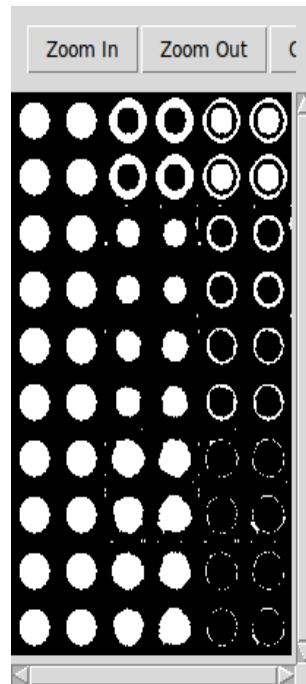


FIGURE 8 –

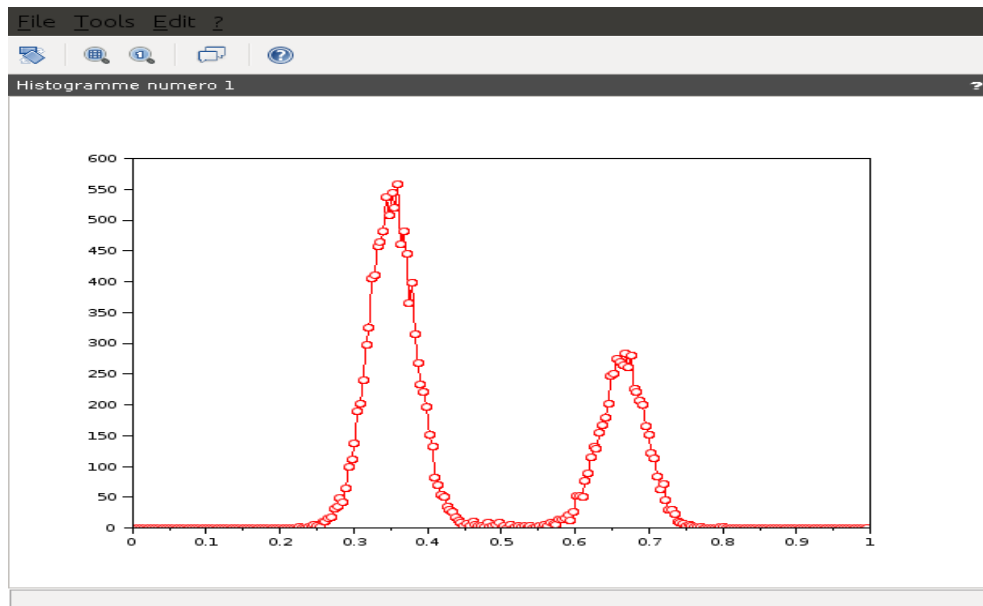


FIGURE 9 – Histogramme calculé à partir de l'image rdf-2-classes-texture-0.png en niveau de gris



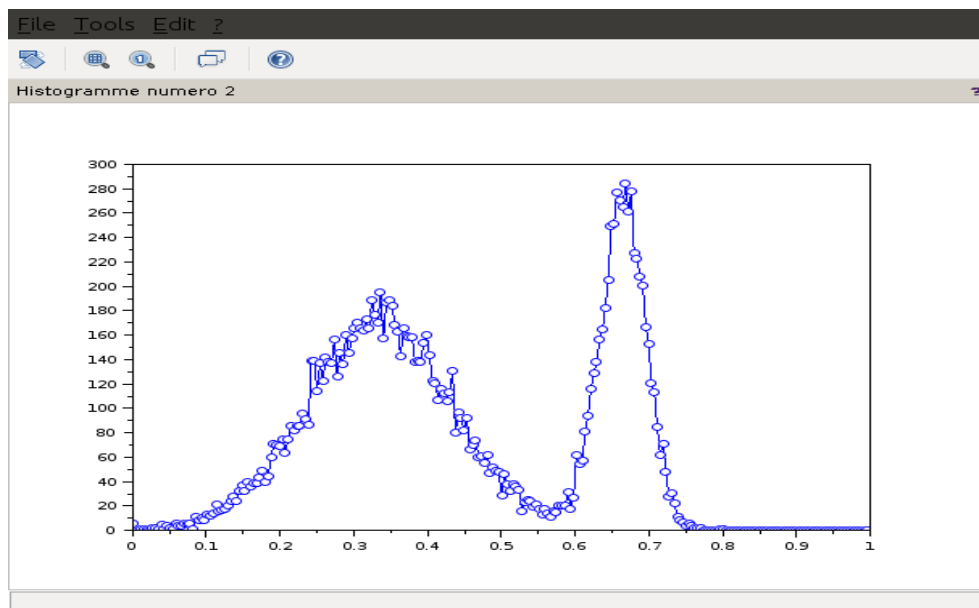


FIGURE 10 – Histogramme calculé à partir de l'image rdf-2-classes-texture-1.png en niveau de gris

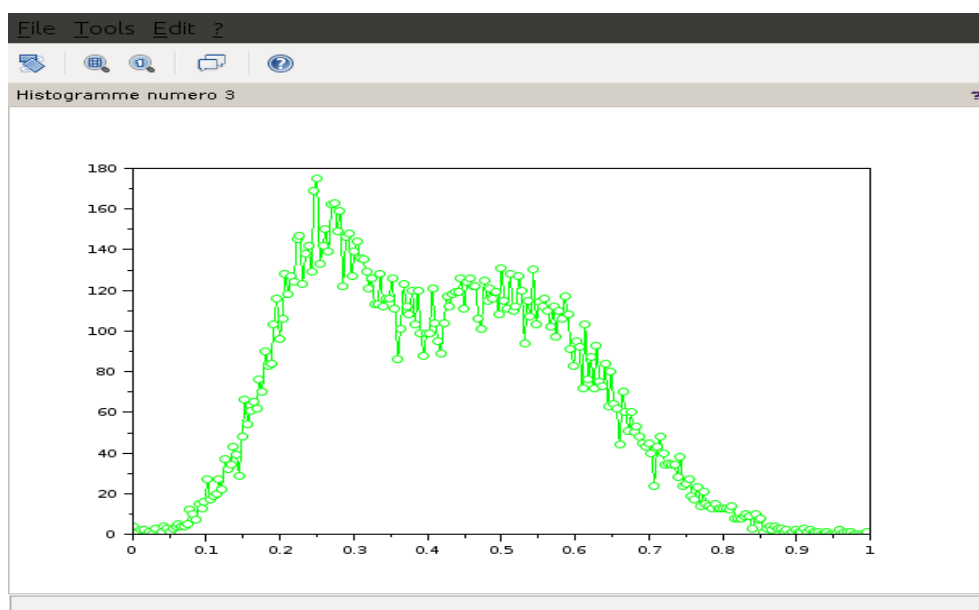


FIGURE 11 – Histogramme calculé à partir de l'image rdf-2-classes-texture-2.png en niveau de gris

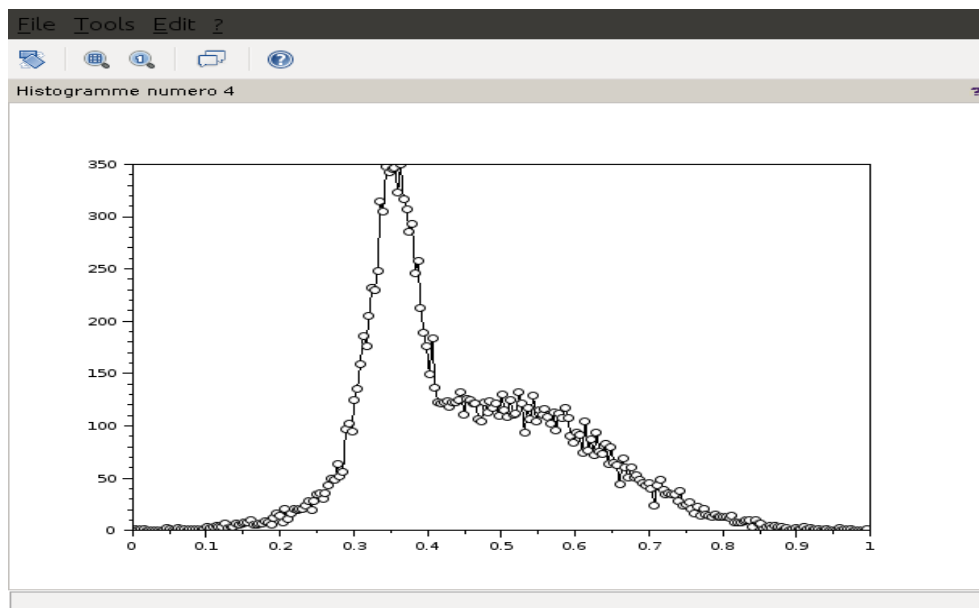


FIGURE 12 – Histogramme calculé à partir de l'image rdf-2-classes-texture-3.png en niveau de gris

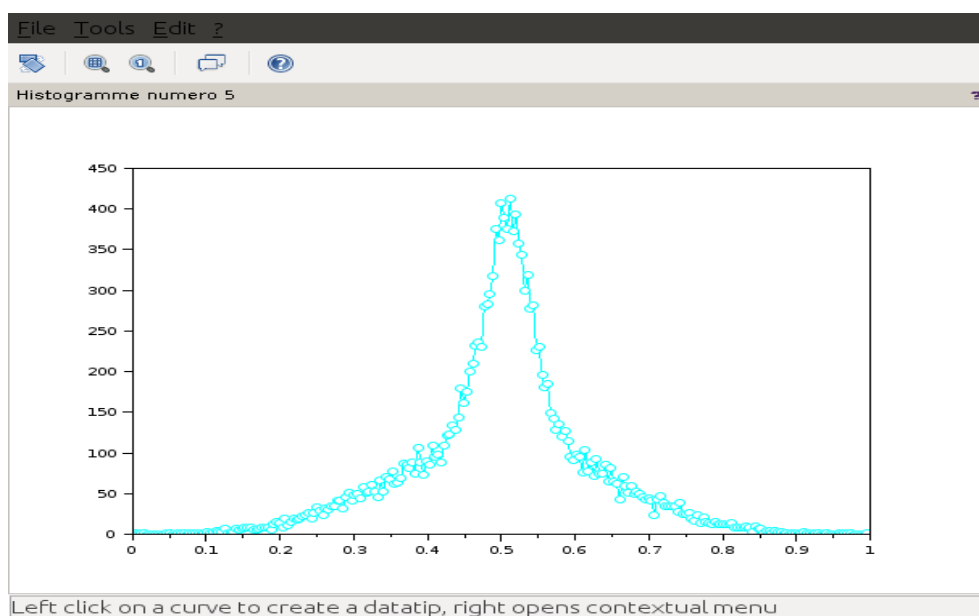


FIGURE 13 – Histogramme calculé à partir de l'image rdf-2-classes-texture-4.png en niveau de gris

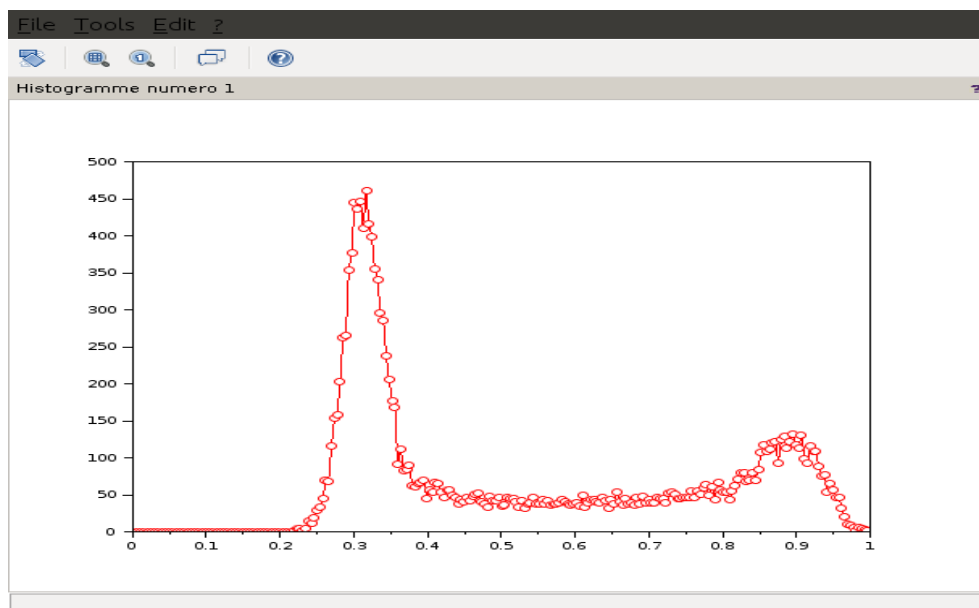


FIGURE 14 – Histogramme calculé a partir de l'image rdf-2-classes-texture-0.png en niveau de texture

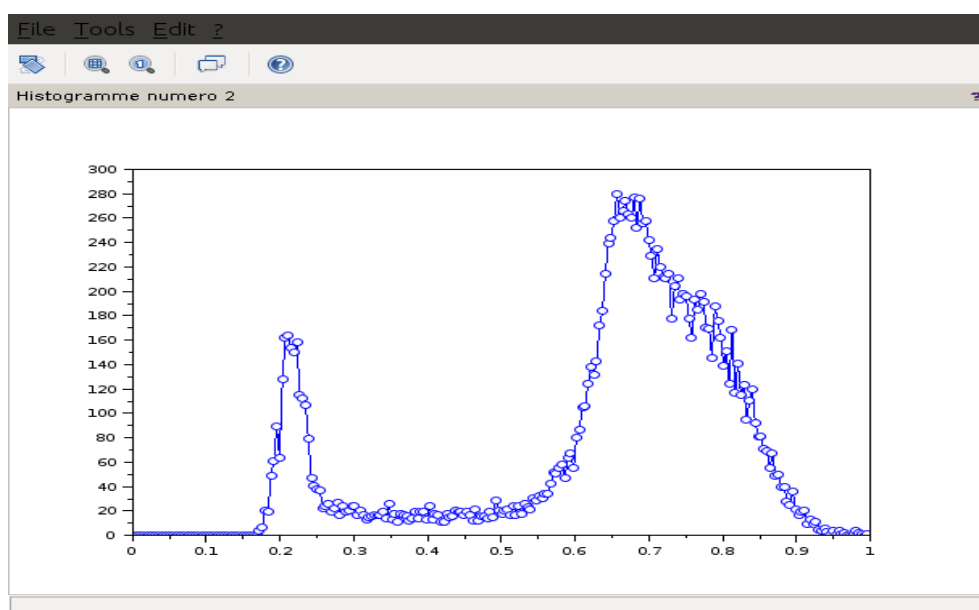


FIGURE 15 – Histogramme calculé a partir de l'image rdf-2-classes-texture-1.png en niveau de texture

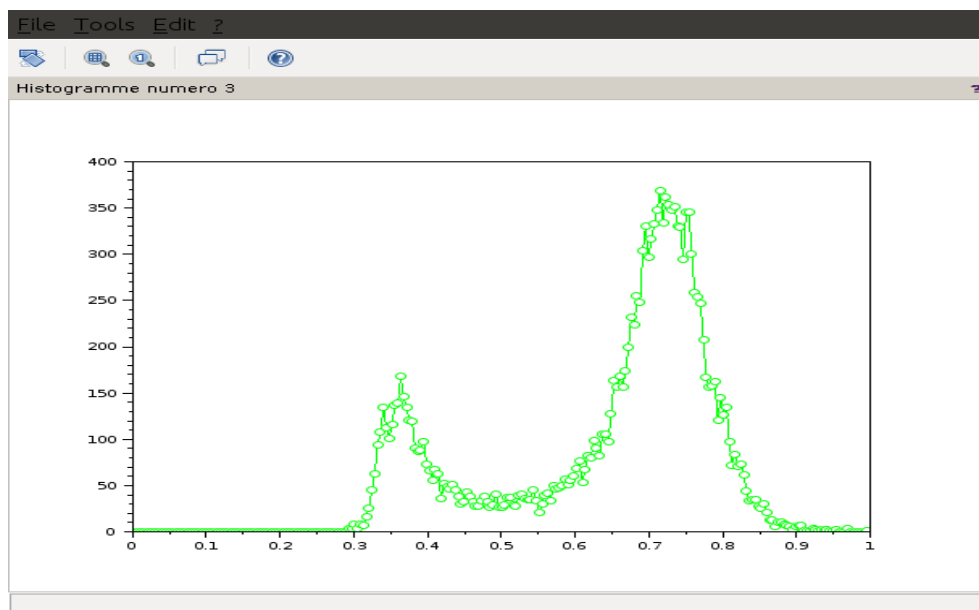


FIGURE 16 – Histogramme calculé a partir de l'image rdf-2-classes-texture-2.png en niveau de texture

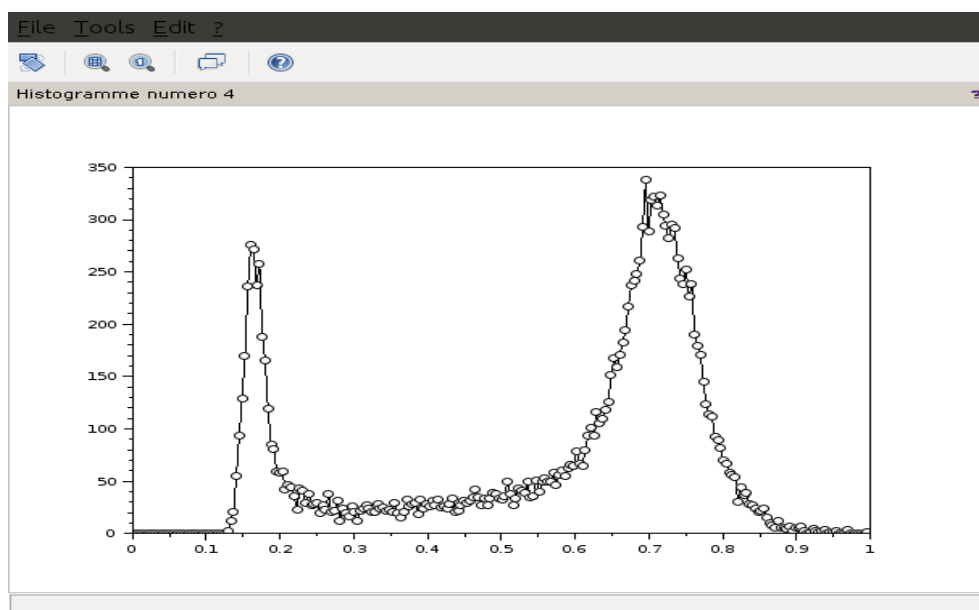


FIGURE 17 – Histogramme calculé a partir de l'image rdf-2-classes-texture-3.png en niveau de texture

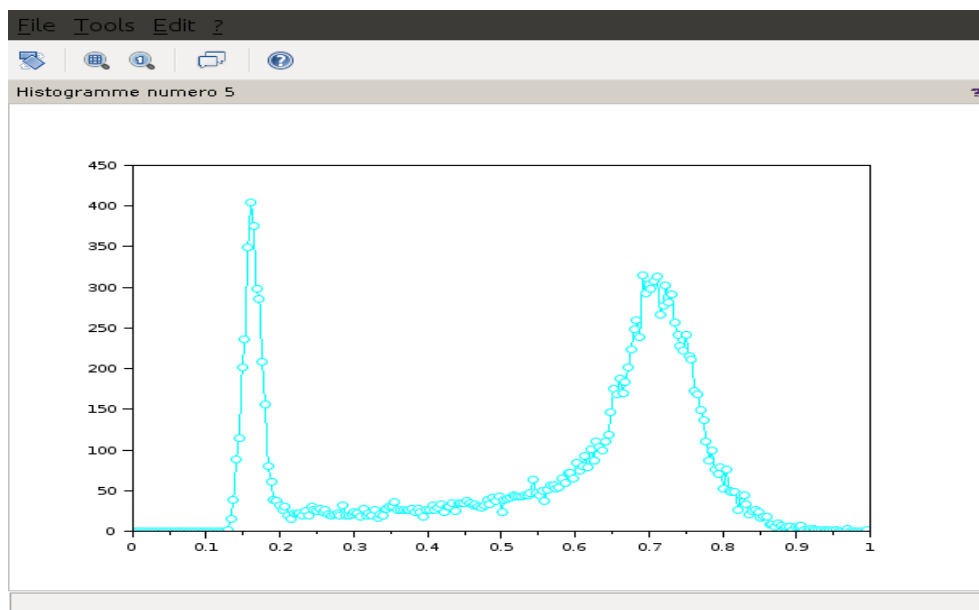


FIGURE 18 – Histogramme calculé a partir de l'image rdf-2-classes-texture-4.png en niveau de texture