TP2 : Codage d'un contour

Benjamin Van Ryseghem
François Lepan

5 février 2013

1 Code Scilab

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

bins permet de définir le pas de l'histogramme. De ce fait, *bins* définit aussi la taille de l'histogramme résultant (*bins*+1). De plus, plus *bins* est grand, plus l'histogramme est précis, c'est à dire, proche de l'image originale.

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 (*c.f* 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 force toutes les valeurs supérieures à ce seuil à devenir 1 (blanc). 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 inférieur à 0.7 devienne 0 (noir) c'est donc pour cela que l'on peut observer des taches noires à l'intérieur des cercles.

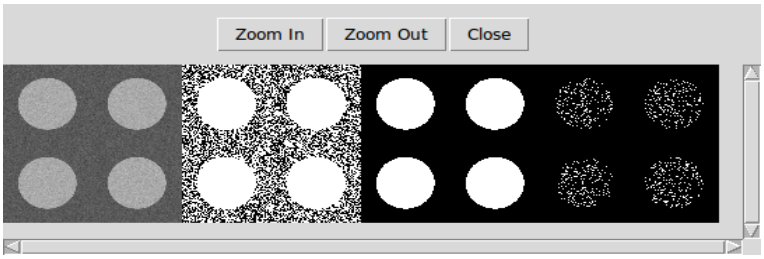

Figure 1 – De gauche à droite : image initiale, image binarisée avec un seuil de 0.35, image binarisée avec un seuil de 0.5, image binarisée avec un seuil de 0.75.

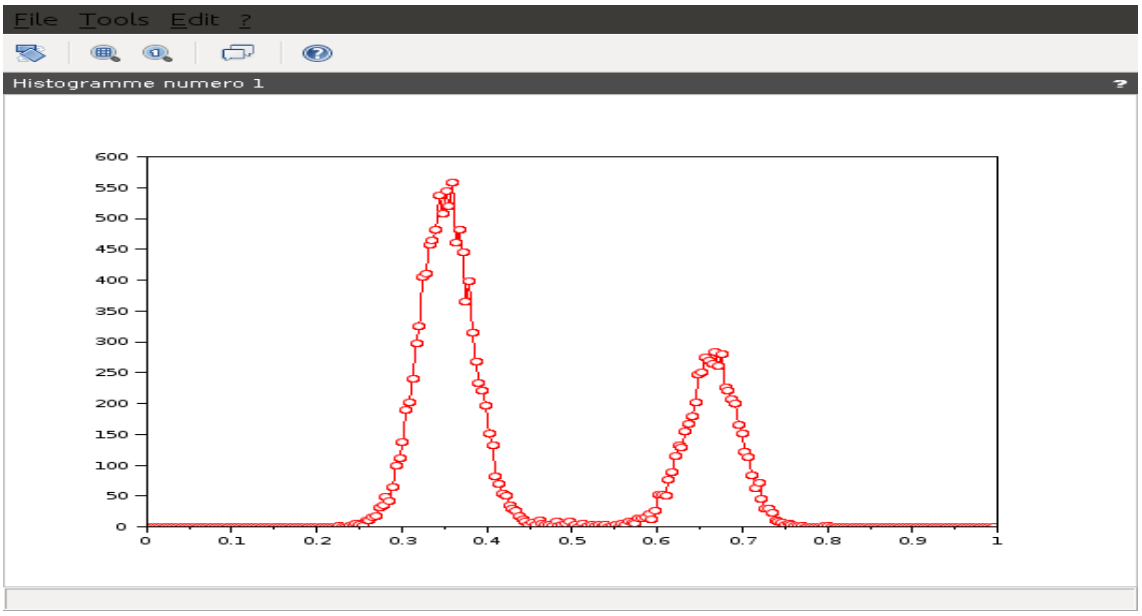

FIGURE 2 – Histogramme de l'image de base (cf : Fig. 1)

1.3 Changer le signe des deux derniers arguments de la fonction *rdfClassifieurLineaire1D* et expliquer le résultat

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

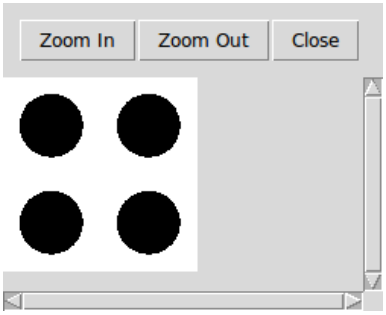

FIGURE 3 – Image résultant de l'inversion des deux derniers arguments de la fonction *rdfClassifieurLineaire1D* sur l'image rdf-2-classes-texture-0.png

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

Le fonction précédente détermine la valeur de retour (0 ou 1) en fonction du calcul de $a \times i + b$. Or changer le signe de a et b revient à changer le sens de l'inégalité précédente.

De ce fait, inverser le signe des arguments revient à échanger le fond avec le premier plan.

Prenons un example : 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 .

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 voulais 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.

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. 11
- rdf-2-classes-texture-1.png Fig. 12
- rdf-2-classes-texture-2.png Fig. 13
- rdf-2-classes-texture-3.png Fig. 14
- rdf-2-classes-texture-4.png Fig. 15

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";
```

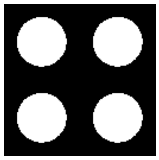

FIGURE 4 – Image référence

```
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

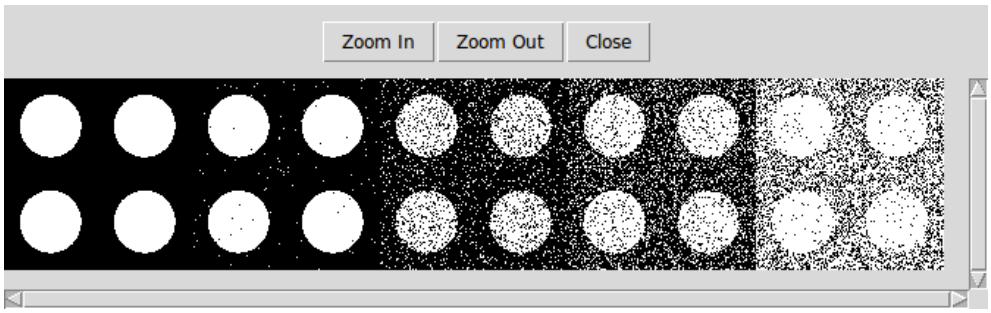

FIGURE 5 – Images texture-0 à 4 binarisées respectivement aux seuil : 0.5, 0.58, 0.31, 0.39, 0.45

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 binarisé 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 résultat ($|$ image binarisée $-$ image de référence $|$).

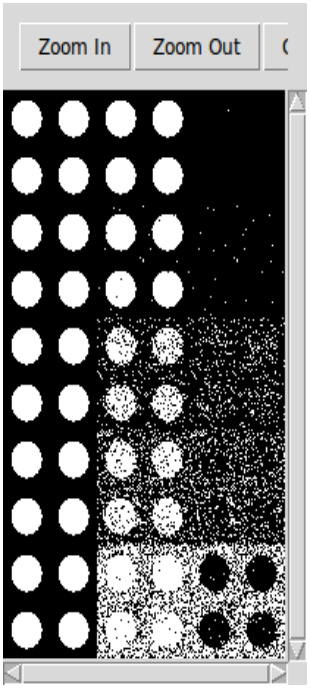

FIGURE 6 –

À la vu 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 meilleurs résultats.

3 Histogramme des niveaux de texture

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

La fonction *rdfTextureEcartType* calcule d'abord la moyenne pour chaque pixel de l'image grâce à la fonction *rdfMoyenneImage*. La fonction *rdfMoyenneImage* a pour but de créer une matrice qui servira de filtre puis l'applique à chacun des pixels de l'image passée en paramètre. Ensuite la fonction *rdfTextureEcartType* calcule la variance suivie de l'écart type.

Cette fonction normalise la matrice puisque le passage au carré modifie grandement les valeurs calculées. Ainsi, on assure que le résultat retourné possède une fourchette de valeurs similaire à celle de l'image passée en argument.

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. 16
- rdf-2-classes-texture-1.png Fig. 17
- rdf-2-classes-texture-2.png Fig. 18
- rdf-2-classes-texture-3.png Fig. 19
- rdf-2-classes-texture-4.png Fig. 20

```
// 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

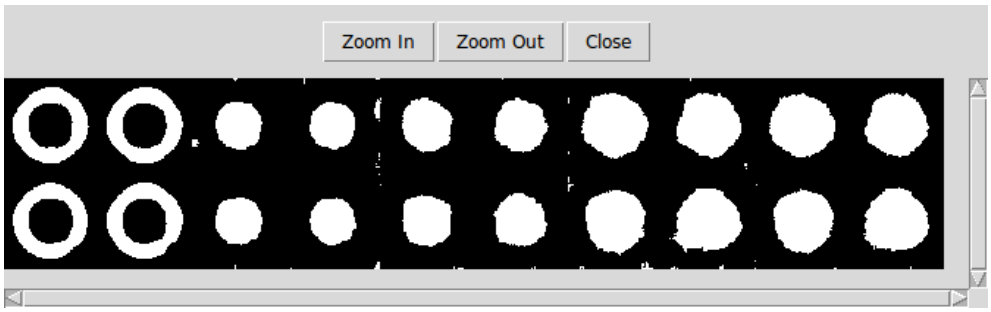

FIGURE 7 – Images texture-0 à 4 en niveau de texture puis binarisées respectivement aux seuils : 0.7, 0.56, 0.57, 0.6, 0.55

Ensuite afin de se rendre compte de la différence entre l'image binarisée 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 en conclure que même si on passe en image de niveau de texture le *bruit* reste important pour certaines images. On observe que pour des images qui ont des textures bien distincts à la base, on obtient plus de bruit car la moyenne des valeurs au niveau de la jonction entre ces textures très différentes donne un nouvelle valeur qui produit du bruit. On peut donc en conclure qu'on ne peut pas utiliser comme seul attribut de chaque pixel son niveau de texture pour la classification.

Au vu des résultats sur la classification en niveau de gris et en niveau de texture on peut se demander si il ne faudrait pas plus d'un classificateur par pixels ?

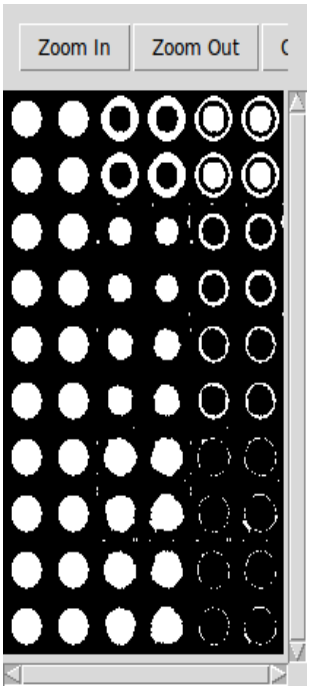

FIGURE 8 – Présentation du résultat idéal (à gauche), le résultat obtenue (au centre), et l'erreur entre les deux (à droite)

4 Histogramme conjoint

4.1 fonction *rdfCalculeHistogramme2D* qui permet de calculer l'histogramme conjoint des deux attributs

```
function hist = rdfCalculeHistogramme2D (image1, bins1, image2, bins2)
  result = zeros (bins2, bins1);
  for y = 1:size (image1, 1)
    for x = 1:size (image1, 2)
        j = int (image1 (y, x) * (bins1 - 1)) + 1;
        i = int (image2 (y, x) * (bins2 - 1)) + 1;
        result (i,j) = result (i,j) + 1;
    end
  end
  // Version logarithmique
  result = log (1 + result);
  // Normalisation a une valeur maximale = 1
  hist = result / max (result);
endfunction
```

Cette méthode consiste principalement à récupérer les valeurs des deux images afin de construire les points de l'histogramme vu comme une image 2D.

5 Classification linéaire à deux dimensions

5.1 Pour chacune des 5 images déterminer les histogrammes conjoints

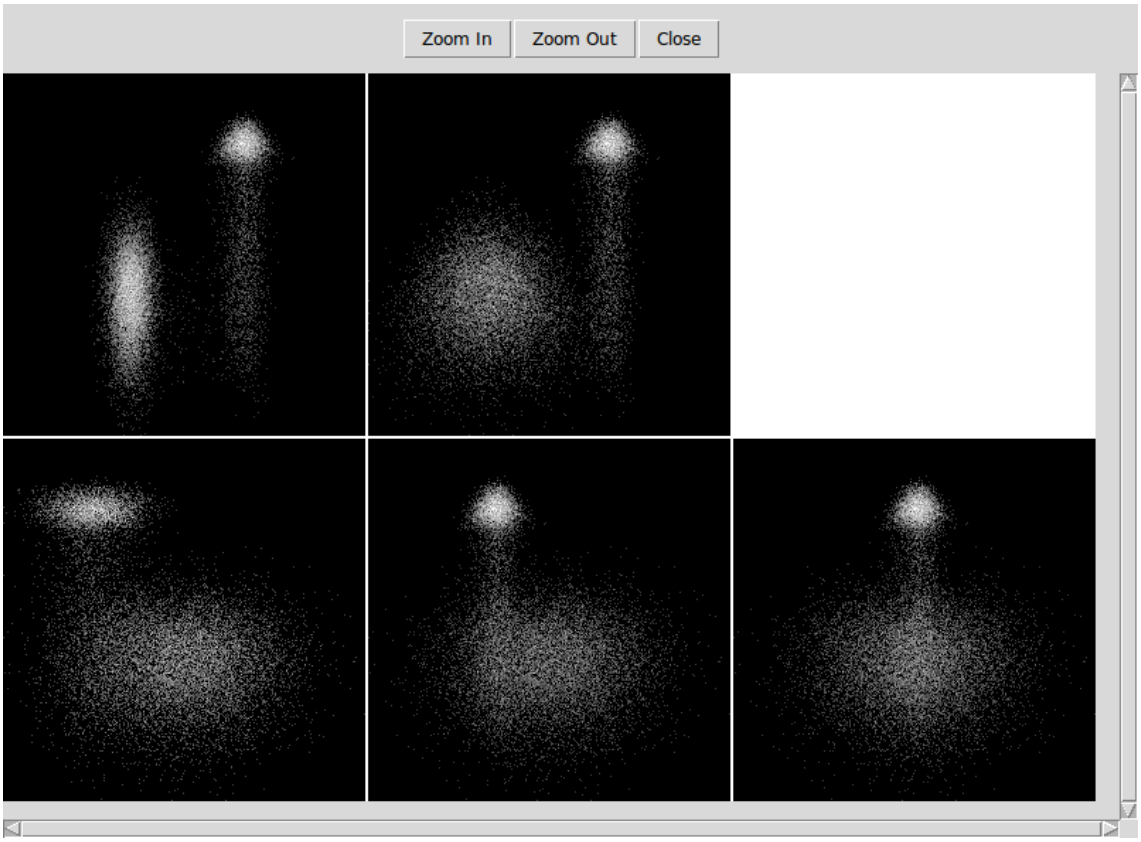

FIGURE 9 – Histogramme conjoints

La figure 9 présente l'histogramme conjoints de chacune des 5 images. On voit se détacher deux zones claires, représentant pour la plus large le fond, et la plus petite les ronds. Au fur à mesure des images les ronds s'assombrissent, ce phénomène est retranscrit sur l'histogramme par la translation de la plus petite zone claire vers la gauche. Sur le dernier histogramme, on remarque que cette zone claire se trouve au centre de l'image, ceci s'expliquant par le fait que la couleur des ronds est proche de la couleur moyenne de l'image.

5.2 Déterminer approximativement la droite qui sépare au mieux les deux régions correspondant : 1) au fond de l'image ; 2) aux 4 disques.

La figure 10 reprend les histogrammes conjoints des 5 images, avec en plus le seuil entre le fond(1) et l'image (2) représenté en rouge.

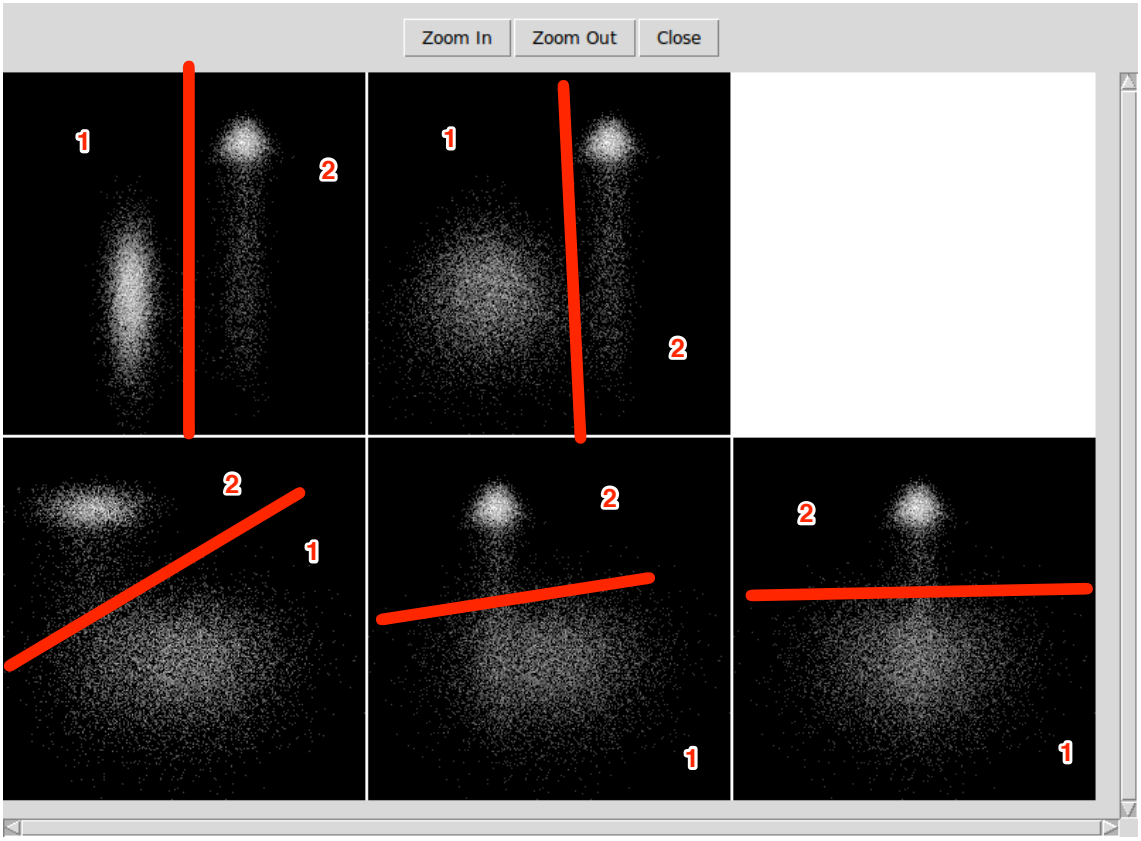

FIGURE 10 – Histogrammes conjoints avec en rouge le seuil séparant le fond de l'image

5.3 Utiliser la fonction rdfClassifieurLineaire2D afin d'obtenir une image binaire à partir d'une classification linéaire 2D

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

Appendices

Histogramme des niveaux de texture

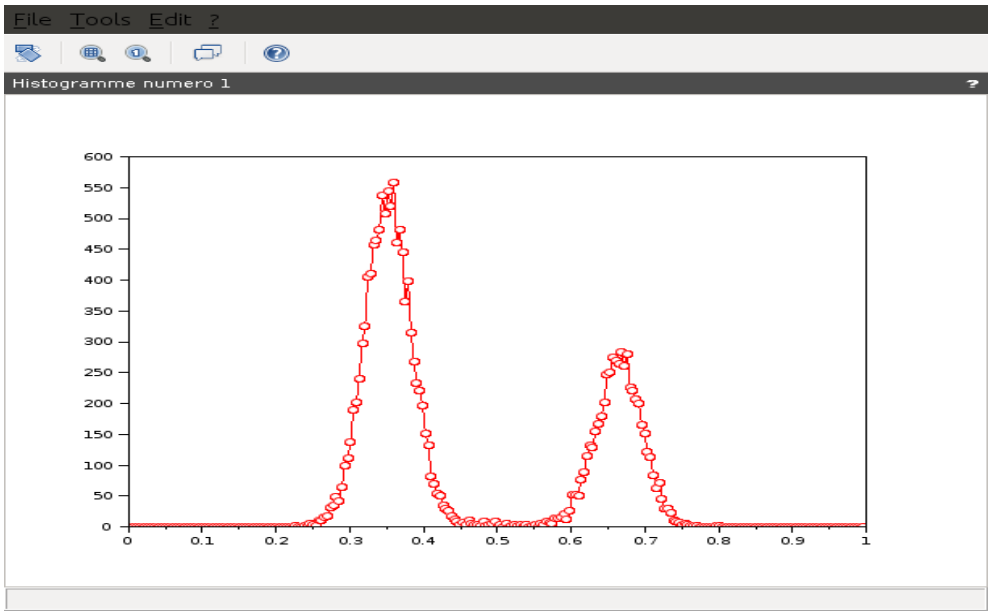

FIGURE 11 – Histogramme calculé à partir de l'image rdf-2-classes-texture-0.png en niveau de gris

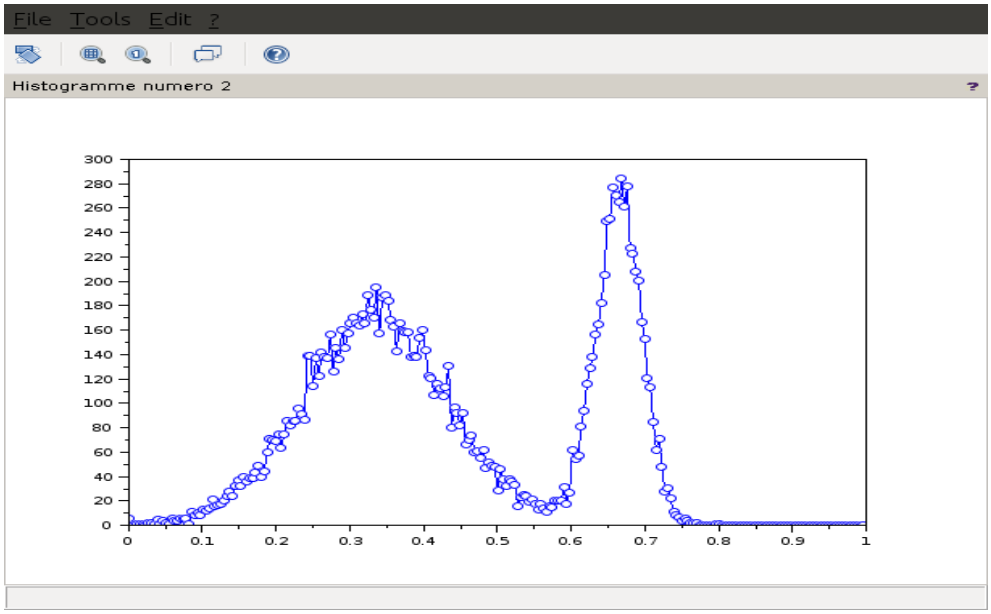

FIGURE 12 – Histogramme calculé à partir de l'image rdf-2-classes-texture-1.png en niveau de gris

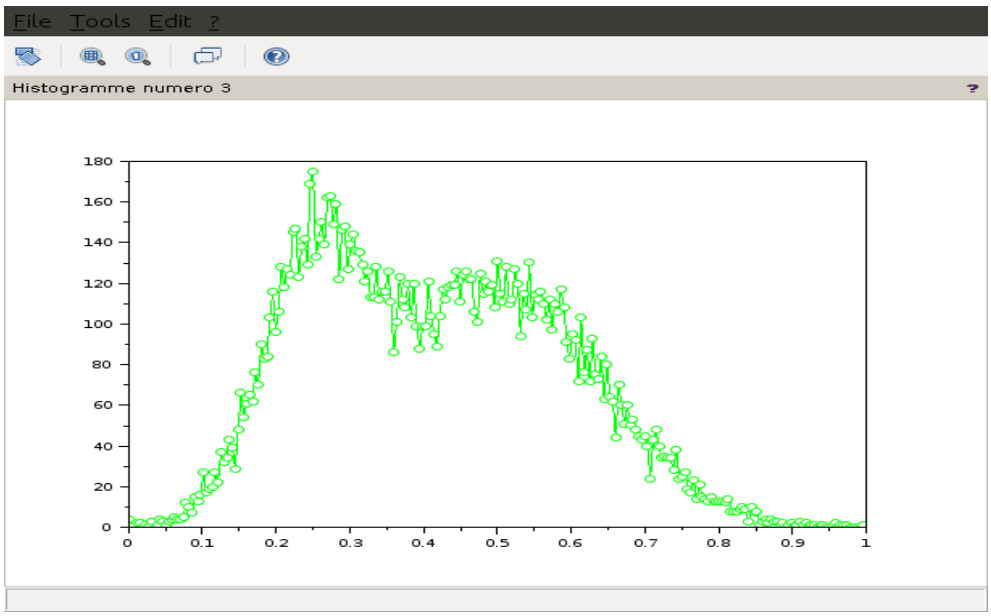

FIGURE 13 – Histogramme calculé à partir de l'image rdf-2-classes-texture-2.png en niveau de gris

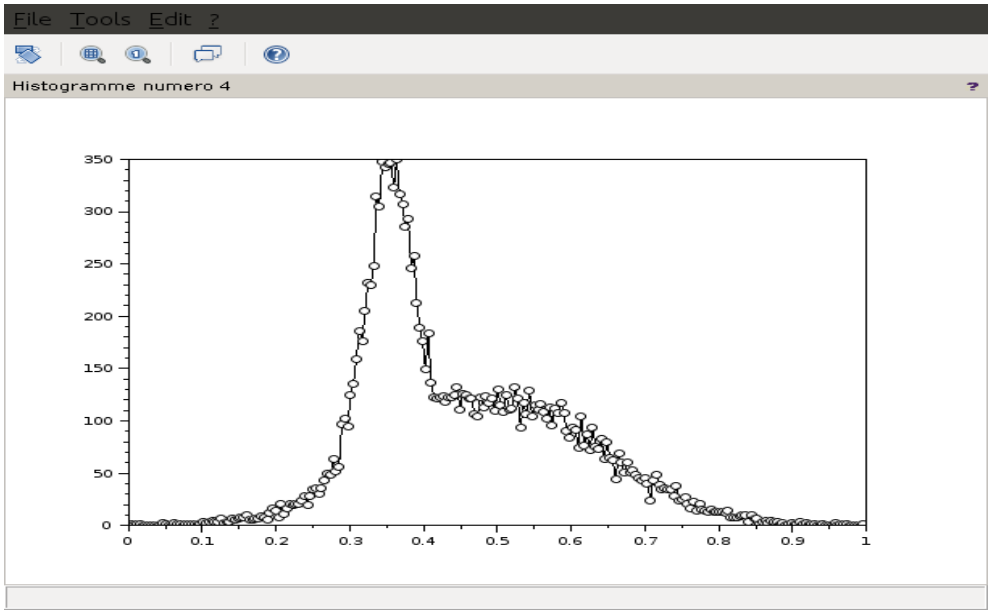

FIGURE 14 – Histogramme calculé à partir de l'image rdf-2-classes-texture-3.png en niveau de gris

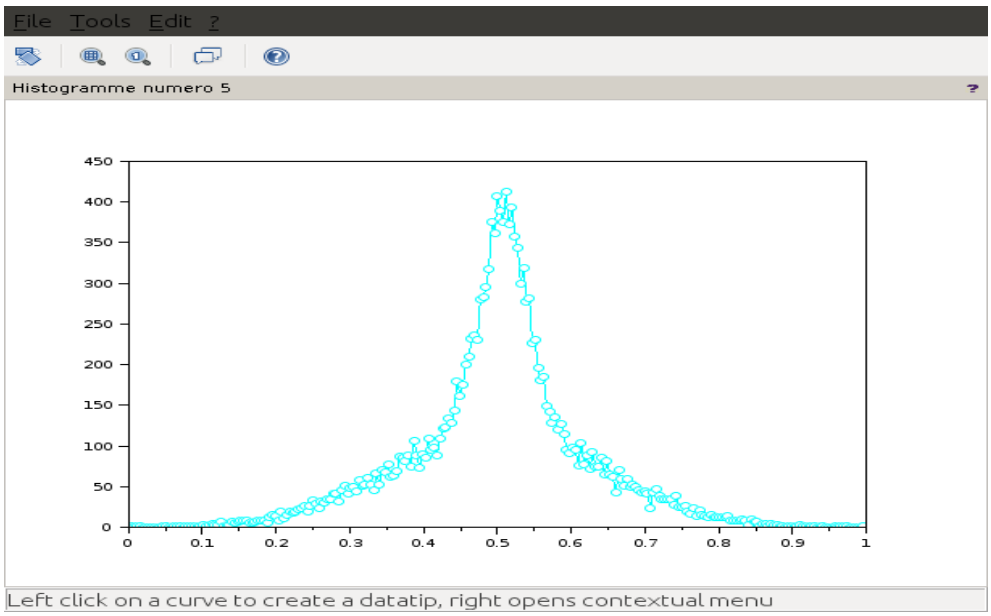

FIGURE 15 – Histogramme calculé à partir de l'image rdf-2-classes-texture-4.png en niveau de gris

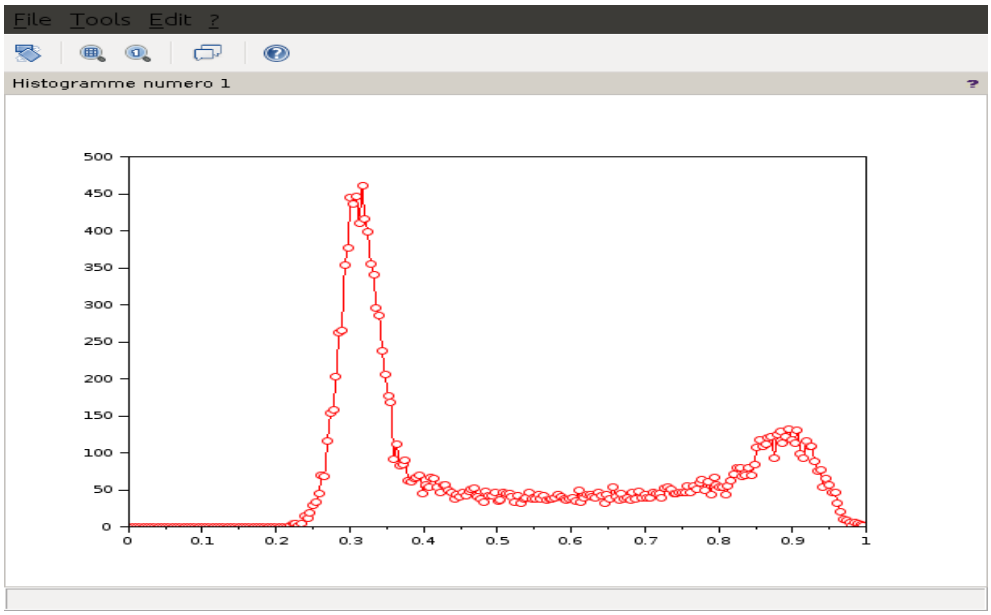

FIGURE 16 – Histogramme calculé a partir de l'image rdf-2-classes-texture-0.png en niveau de texture

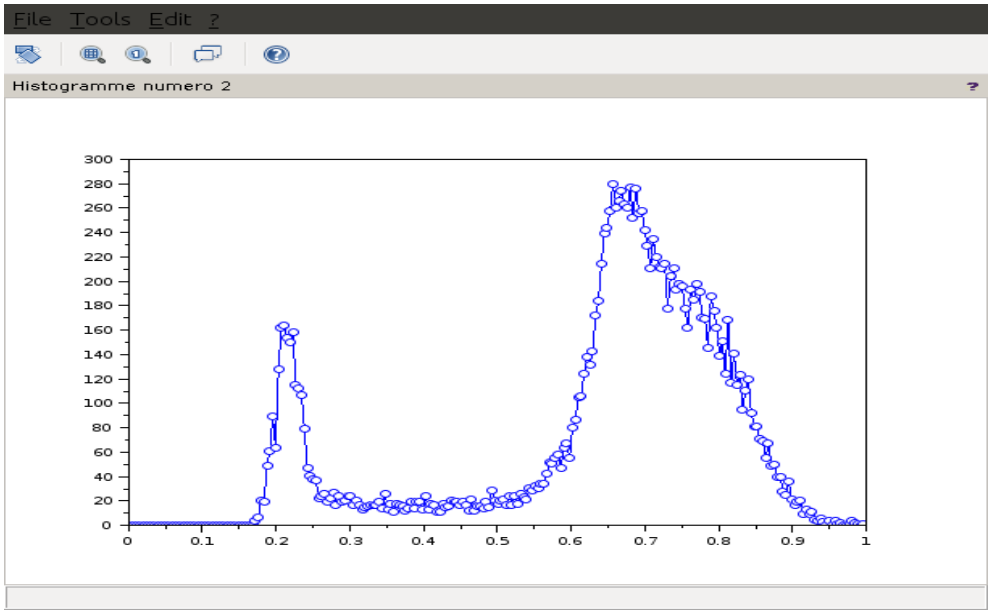

FIGURE 17 – Histogramme calculé a partir de l'image rdf-2-classes-texture-1.png en niveau de texture

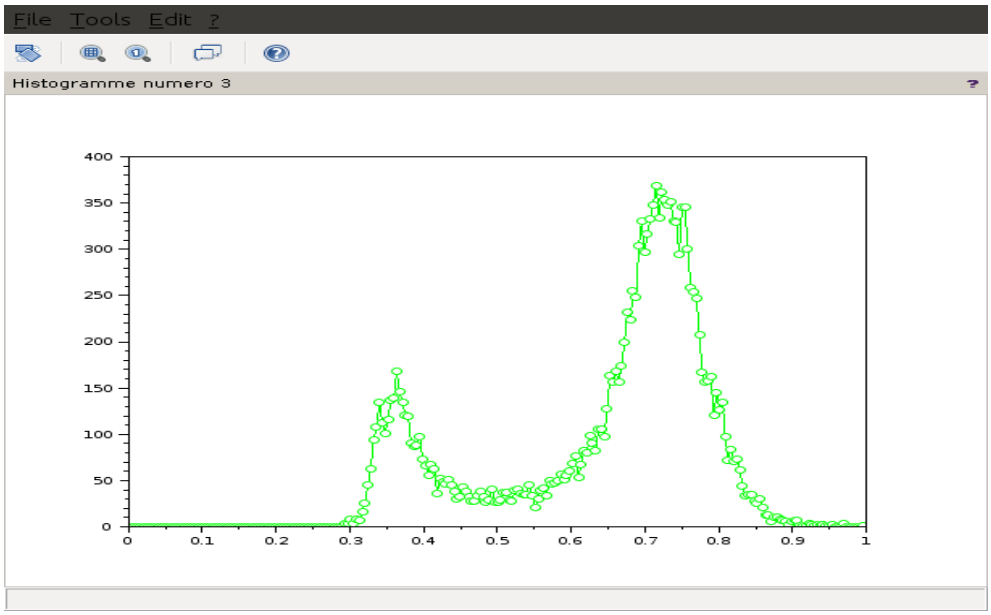

FIGURE 18 – Histogramme calculé a partir de l'image rdf-2-classes-texture-2.png en niveau de texture

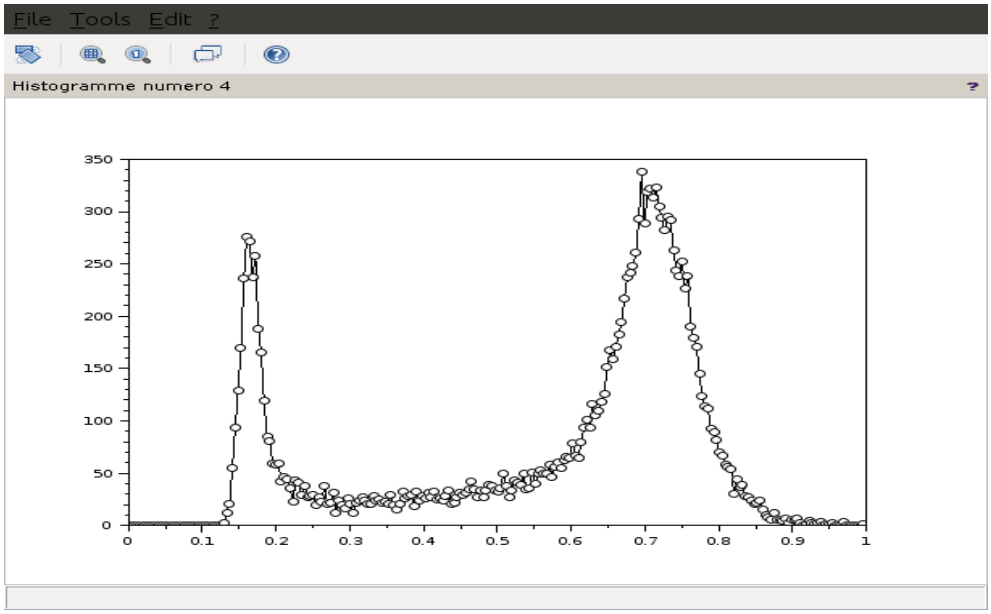

FIGURE 19 – Histogramme calculé a partir de l'image rdf-2-classes-texture-3.png en niveau de texture

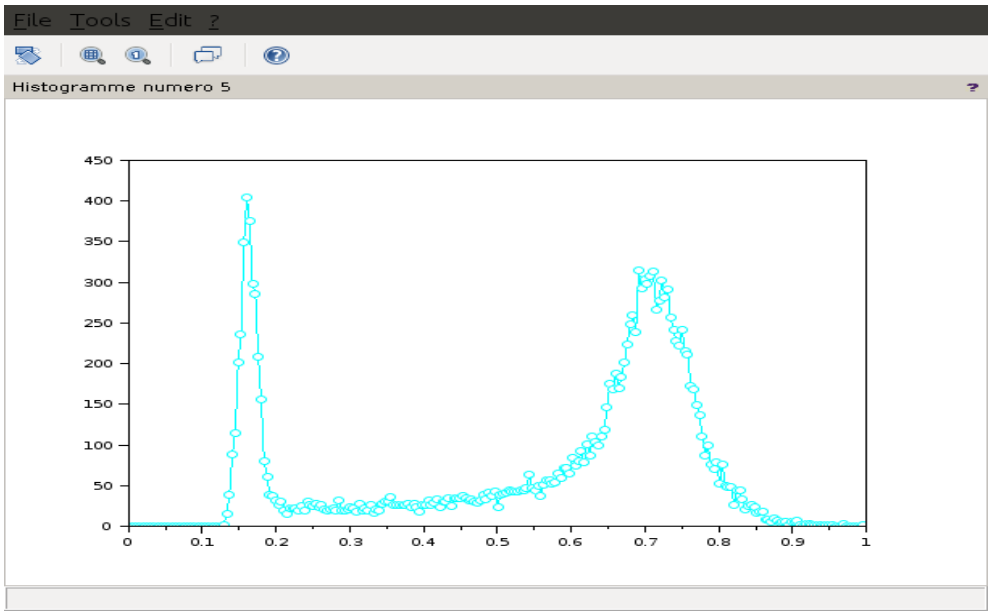

FIGURE 20 – Histogramme calculé a partir de l'image rdf-2-classes-texture-4.png en niveau de texture