

Compte rendu TP3 Rdf

Antoine Canda

Arthur d’Azémar

28 février 2017

1 Introduction

Le but de ce TP est de découvrir différentes méthodes de binarisation d’image, ici notamment texturé, pour mettre en lumière les difficultés que l’on peut rencontrer. On commencera par binariser selon les niveaux de gris puis selon les textures avant de combiner les deux méthodes dans une binarisation selon plusieurs attributs.

2 Code R

2.1 A quoi correspond l’argument nbins utilisé pour calculer l’histogramme des niveaux de gris ? Modifier sa valeur (autre que 256) et commenter les résultats.

Nbins est le nombre de niveaux de gris différents disponibles.

En modifiant la valeur de Nbins, on modifie le nombre de niveaux de gris disponibles. Il y a donc plus ou moins de valeurs sur l’abscisse de l’histogramme.

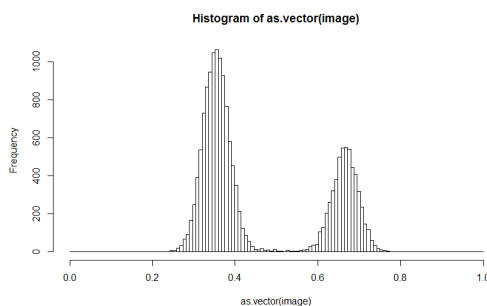


FIGURE 1 – 128 niveaux de gris

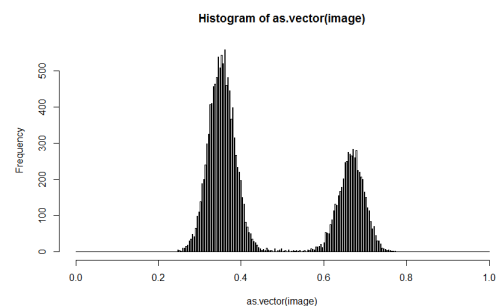


FIGURE 2 – 512 niveaux de gris

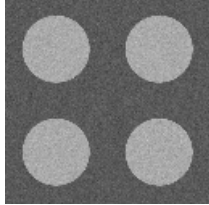
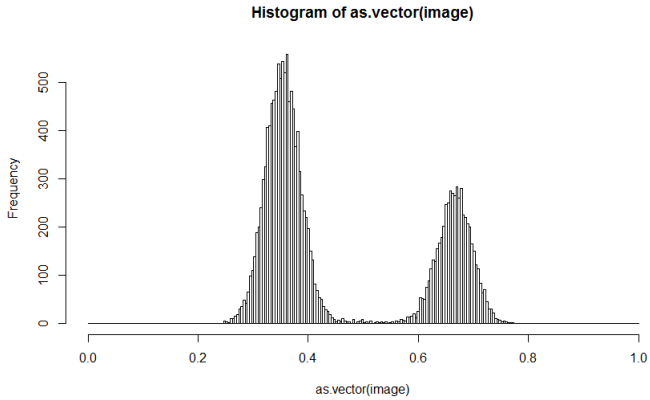
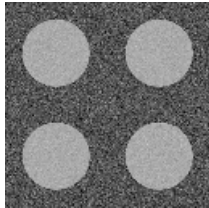
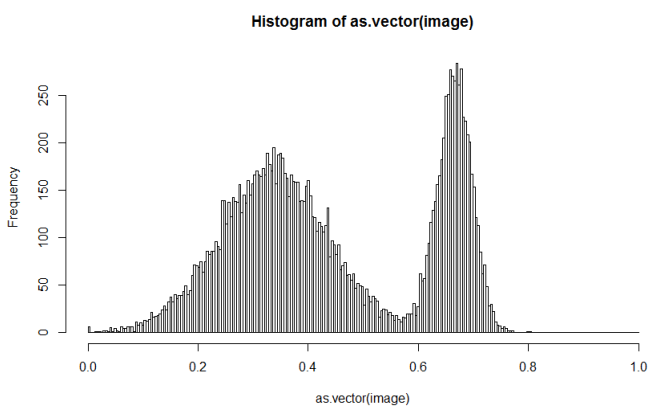
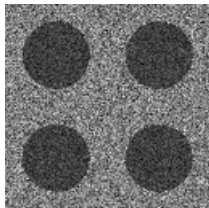
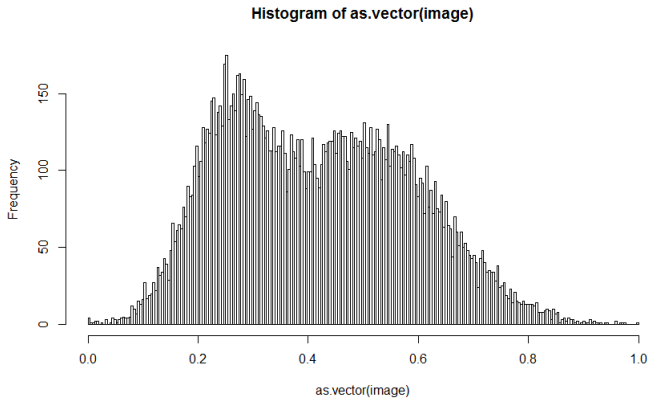
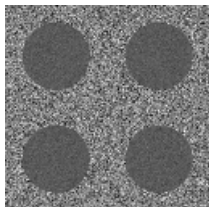
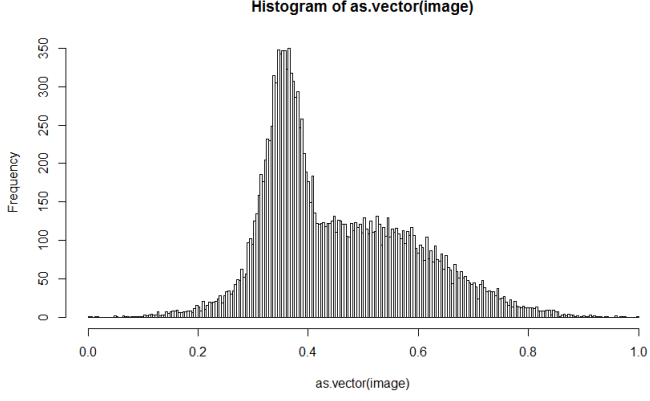
2.2 modification du seuil de binarisation

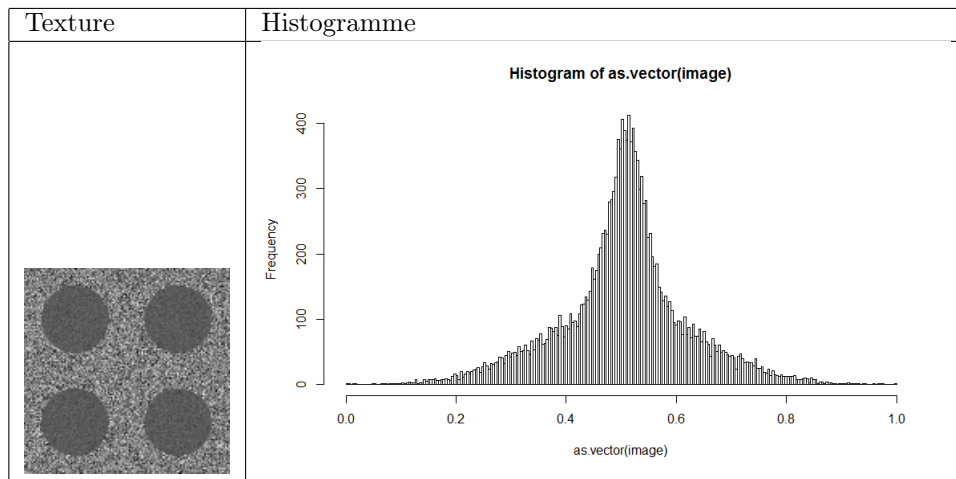
En modifiant la valeur de seuil, on obtient une binarisation plus ou moins précise.

3 Histogramme des niveaux de gris

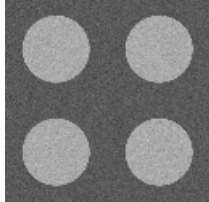
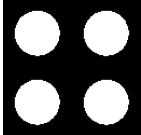
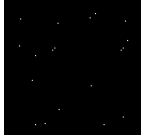
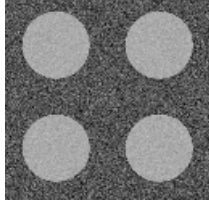
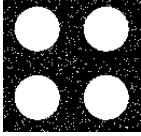
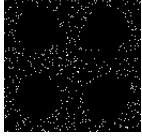
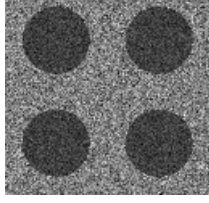
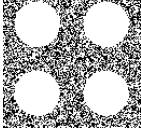
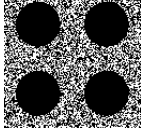
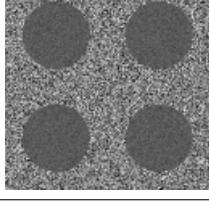
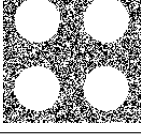
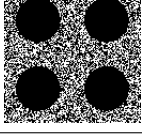
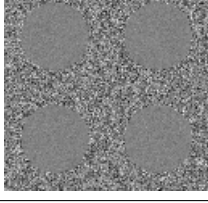
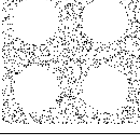
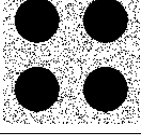
3.1 Histogramme des niveaux de gris des différentes images

On a utilisé la fonction hist sur chacune des quatre textures afin d’obtenir l’histogramme des niveaux de gris de la fonction. On obtient les histogrammes suivants :

Texture	Histogramme
	<p>Histogram of as.vector(image)</p> 
	<p>Histogram of as.vector(image)</p> 
	<p>Histogram of as.vector(image)</p> 
	<p>Histogram of as.vector(image)</p> 



3.2 Détermination empirique de seuils pour binariser les images

Texture	Seuil	Image binaire	Image différence	Taux d'erreur
	0.5			0.11%
	0.5			3.29%
	0.51			35.31%
	0.5			33.14%
	0.35			56.70%

Dans le tableau ci-dessus on présente les résultats obtenus dans la recherche de binarisation après un seuillage trouvé de manière empirique.

Pour chaque texture, on présente le seuil trouvé, l'image binaire obtenue puis on a calculé la

différence de pixel entre chaque image binarisé obtenue et le masque de référence à la base sous forme d'image pour avoir un retour visuel et on a compté le nombre de pixel qui étaient différent pour obtenir un pourcentage de différence.

3.3 Conclusion

On remarque que mis à part les textures 0 et 1 qui possèdent des histogrammes par ailleurs plus facilement exploitable que les 3 autres textures, on a un pourcentage d'erreurs entre l'image référence et l'image obtenue important.

Il est donc nécessaire d'utiliser un autre attribut pour pouvoir segmenter les images.

4 Histogramme des niveaux de textures

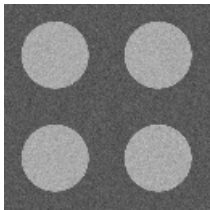
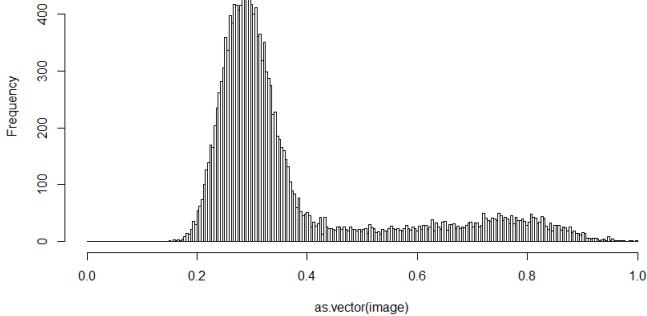
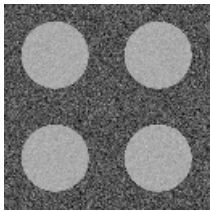
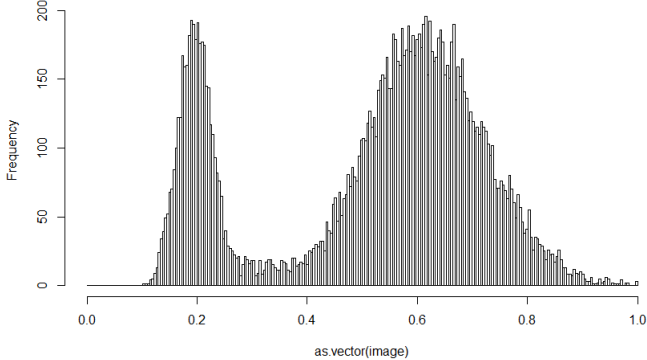
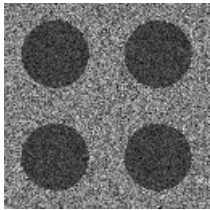
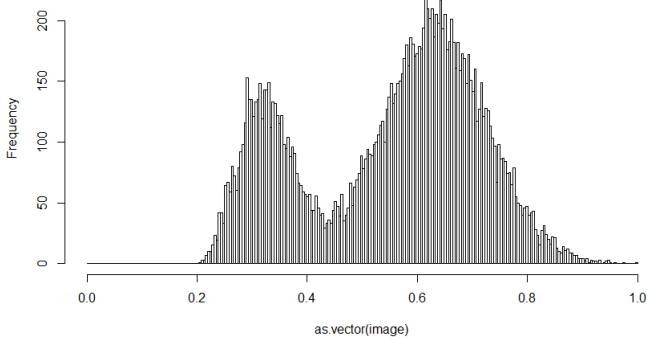
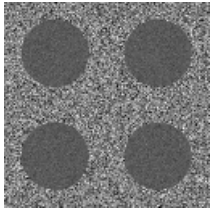
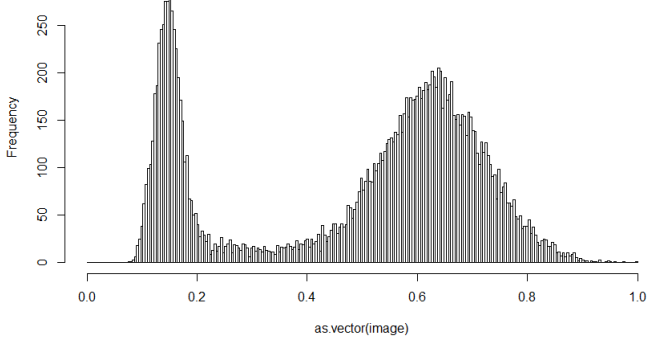
4.1 fonctionnement de la fonction `rdfTextureEcartType`

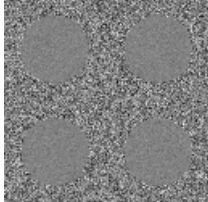
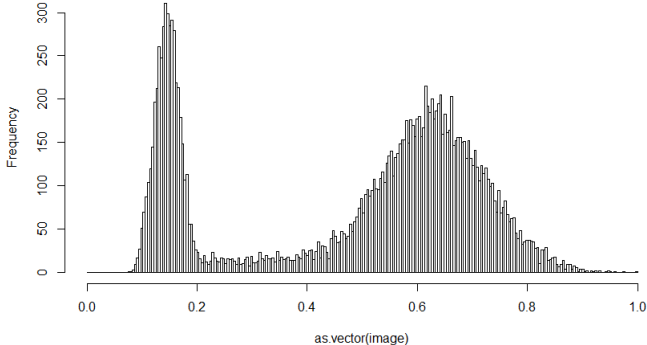
La fonction `rdfTextureEcartType` va commencer par calculer pour chaque pixel, le carré de son niveau de gris moins la moyenne de des niveaux de gris des pixels voisins. Ces valeurs seront stockées dans la matrice `carre`. Pour calculer l'écart type, on calcule la racine carré de la moyenne des valeurs de la matrice `carre`. (ces valeurs sont stockées dans la matrice `ecart`). Enfin, on normalise les valeurs en divisant les valeurs de la matrice `ecart` par la valeur maximale de cette matrice.

On fixe la dimension du voisinage de calcul dans la fonction `rdfMoyenneImage`. La taille correspond au rayon de la zone de voisinage.

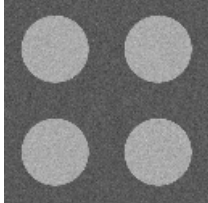
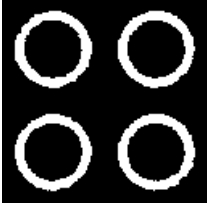
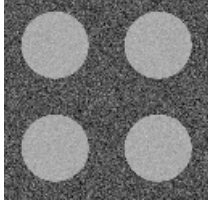
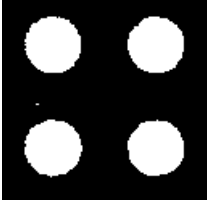
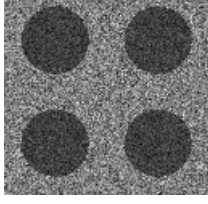

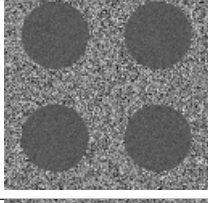
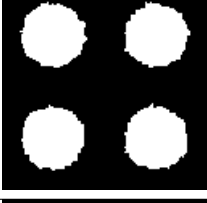
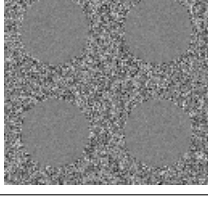
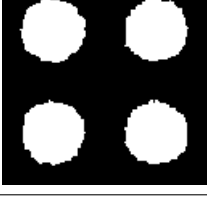
On normalise l'image écart-type car on obtient des valeurs supérieurs à celle de base. La normalisation permet de revenir à des valeurs correspondantes.

4.2 seuil de binarisation des textures

Texture	Histogramme	Seuil de binarisation (%)
	<p>Histogram of as.vector(image)</p> 	0.5
	<p>Histogram of as.vector(image)</p> 	0.3
	<p>Histogram of as.vector(image)</p> 	0.46
	<p>Histogram of as.vector(image)</p> 	0.35

Texture	Histogramme	Seuil de binarisation (%)
	<p>Histogram of as.vector(image)</p> 	0.3

4.3 binarisation des textures

Texture	Binarisation	taux d'erreur (%)
		34,17
		9.80
		6.27
		3.90
		4.06

4.4 conclusion

Quand on regarde les différents résultats obtenus, que ce soit les images ou les pourcentages d'erreurs, on peut observer que dans certains cas on obtient des résultats bien meilleurs à ce que

l'on a eu avant mais aussi bien plus mauvais. On peut par exemple prendre le cas de la texture 0 qui passe d'un taux d'erreur quasiment nul à un taux d'erreurs de plus d'un tiers de ses pixels. Mais les textures 2, 3 et 4 passent d'un taux d'erreur de plus de 30% à moins de 6.3%. L'image obtenu de la texture 0 montre que la texture a crée un bruit important. On en conclut donc que bien qu'intéressant dans certains cas, utiliser uniquement le niveau de texture comme attribut n'est pas suffisant mais reste intéressant. La combinaison de plusieurs attributs peut être permettre de combiner le meilleur des différents cas et obtenir une binarisation plus proche de celle attendue.

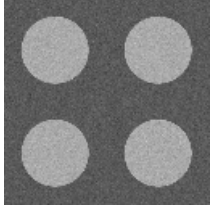
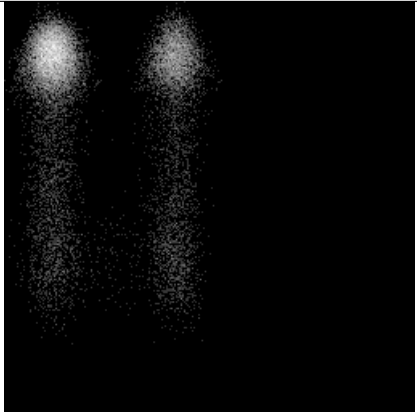
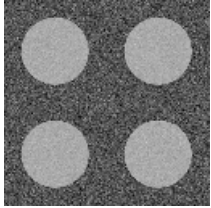
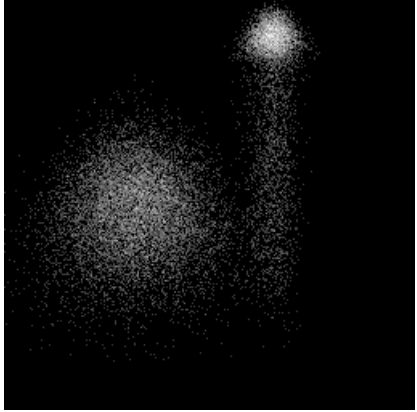
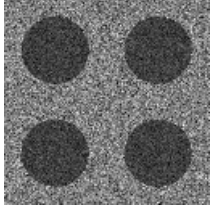
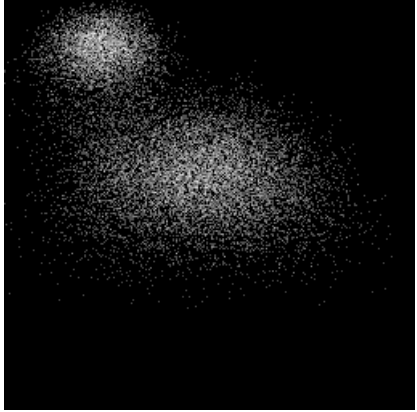
5 Histogramme conjoint et classifieur 2D

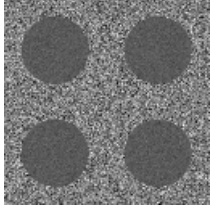
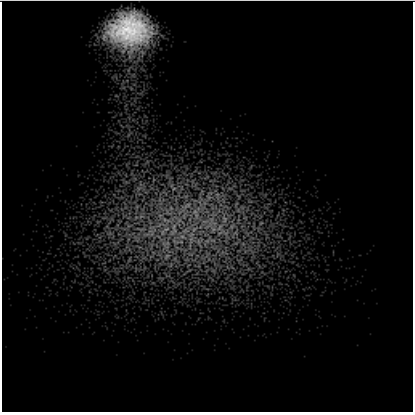
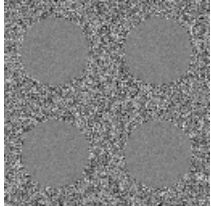
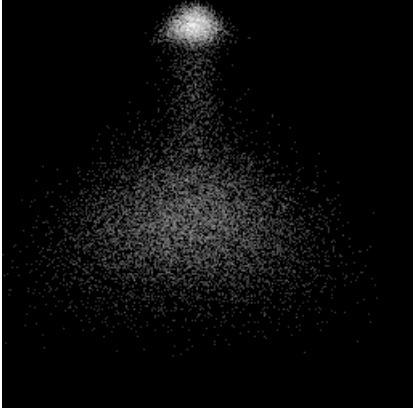
5.1 Histogramme conjoint

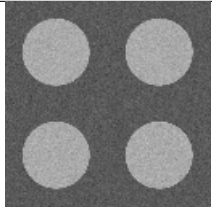
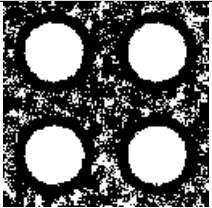
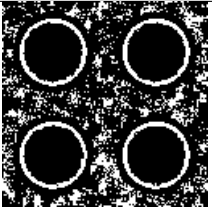
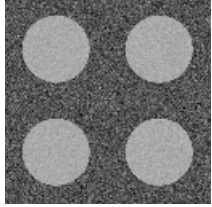
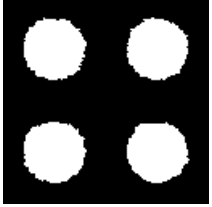
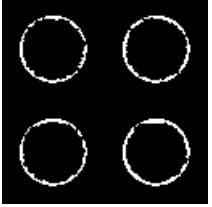
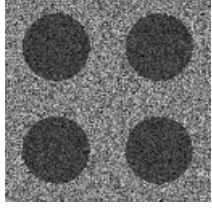
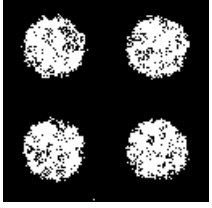
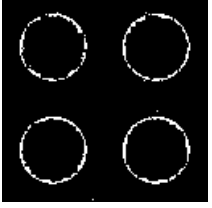
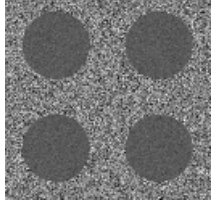
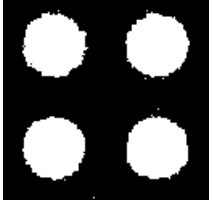
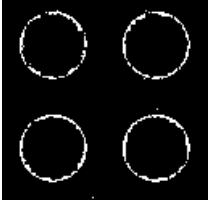
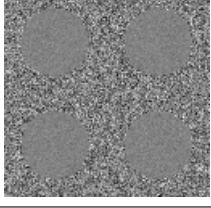
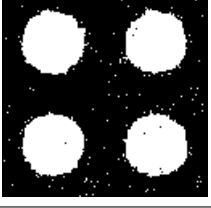
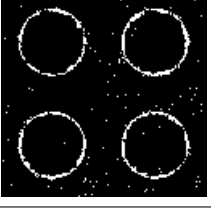
On commence par calculer les intervalles entre les valeurs x et y de la matrice pour les deux images avec la fonction `findInterval`. Ensuite on va créer une matrice de 256*256 pour lequel on va calculer le nombre d'occurrences des valeurs de niveaux de gris avec la fonction `table`. Puis, on la complète avec la valeur 0. Enfin on utilise la fonction `log` pour afficher les zones de faible surface de l'histogramme soit présente et on normalise pour afficher sous forme d'image.

5.2 Classification linéaire à deux dimensions

On a commencé par calculer les histogrammes conjoints des différentes textures et on a obtenu les résultats suivants :

Texture	Histogramme 2D
	
	
	

Texture	Histogramme 2D
	
	

Texture	Binarisation	diff des images	taux d'erreur (%)
			21.82
			5.32
			10.88
			4.38
			5.29

Nous avons essayé de faire une fonction qui a partir de la valeur d'un pixel en position x,y (on considère les deux images qui ont permis d'obtenir l'histogramme conjoint) que l'on nommera i et j , avec trois flottants a , b et c qui sont des coefficients valant 1 ou -1 pour a et b et le seuil pour c d'obtenir une valeur selon la formule $ai + bj + c$ que l'on compare à 0. Si elle est plus grande on attribue au pixel la valeur 1 sinon la valeur 0.

Nous obtenons des résultats qui sont un minimum correct mais nous ne sommes absolument pas certains d'avoir pour autant d'avoir fait un classifieur correct. Et les résultats obtenus nous font penser qu'on a peut être faux.

6 Conclusion

On peut voir que les résultats varient fortement d'une méthode à l'autre pour la binarisation d'une image selon la texture qu'elle présente. Néanmoins on remarque que globalement la classification avec plusieurs attributs fonctionnent un peu mieux que celle ne considérant que la texture qui elle même a de meilleure résultat sur nos images que le seul niveau de gris. Après il est important de noter que selon l'image la classification à partir d'un seul attribut a de meilleure résultat qu'une autre méthode qui peut considérer un autre attribut unique ou une combinaison d'attributs comme la texture 0 et la méthode basé sur les niveaux de gris qui a un taux d'erreur quasiment nul.

Il est donc nécessaire si on souhaite une binarisation parfaite d'une seule image de considérer qu'elle est la meilleure méthode pour obtenir le meilleur résultat.