2.1 Introduction :

Dans le domaine du traitement d'image, la notion d'analyse multirésolution apparait souvent. Elle incorpore et unifie des techniques développées pour différentes disciplines dont le codage en sous-bandes de signaux, les filtres miroirs en quadrature en reconnaissance de la parole et l'extraction de primitives dans les représentations pyramidales d'images [32]. L'interprétation d'une image revient à détecter les primitives visuelles qui la composent. Ces primitives peuvent être entre autres des arêtes, des contours ou des coins et admettent des dimensions variées tels que la longueur, la largeur, la profondeur, l'épaisseur, etc. Pour détecter ces dernières, nous avons besoin de plusieurs opérations de bas niveau [33] comme le filtrage, la dérivée, le sous-échantillonnage, etc. Durant les dernières décennies, plusieurs chercheurs ont travaillé sur les transformées multirésolutions pour remédier à la limite majeure de la transformée de Fourier (FFT) qui ne garde que l'information fréquentielle et perd l'information spatiale de l'image. Les transformées multirésolutions permettent, non seulement l'interprétation de l'information fréquentielle, mais aussi sa localisation dans l'image. Les transformées multirésolutions visent essentiellement à assurer les caractéristiques suivantes :

La multirésolution : La représentation devrait permettre aux images d'être successivement approximées, d'une résolution grossière à une résolution fine.

La localisation : Les éléments de base dans la représentation doivent être localisés dans le domaine spatial et fréquentiel.

L'échantillonnage critique : Pour certaines applications (ex. la compression), la représentation devrait constituer une base ou un cadre avec une redondance de petite taille.

La directionnalité : La représentation doit contenir des éléments de base orientés à une variété de directions, beaucoup plus que les quelques directions qui sont offertes par les ondelettes séparables.

L'anisotropie : Pour capturer des contours lisses dans les images, la transformée devrait avoir des éléments de base qui permettent de faire la recherche directionnelle de contours. Ceci est réalisé en utilisant, lors de la décomposition, une variété de formes allongées avec différentes directions.

La transformée en ondelettes :

La transformée en ondelettes discrète (DWT) bidimensionnelle permet la représentation d'une image dans le domaine spatio-fréquentiel. Elle est basée sur l'analyse multirésolution qui décompose hiérarchiquement un signal (image) en deux parties. La première partie, de basse fréquence, est une sorte de moyenne du signal d'origine communément appelée

image d'approximation. La seconde partie est un ensemble de sous-bandes constituant les détails de l'image ; à chaque niveau de résolution, les détails sont organisés sous forme de 3 sous-bandes orientées horizontalement (H), verticalement (V) et diagonalement (D). Ces détails sont communément appelés les coefficients ondelettes. À l'origine, les ondelettes ont été introduites par Alfred Haar en 1909. Elles ont été appliquées pour représenter des signaux monodimensionnels. Mallat [34] a étendu l'application de la transformée en ondelettes aux images (ou signaux bidimensionnels). Il a ainsi introduit l'algorithme rapide de décomposition/reconstruction par ondelettes. Cet algorithme est récursif et se base essentiellement sur deux opérations :

Le filtrage :convolution du signal avec un filtre passe-bas (h_0) ou un filtre passe-haut (g_0).

Sous-échantillonnage :réduction du nombre d'échantillons du signal. En effet, un sous-échantillonnage horizontal (1 : 2) de l'image revient à éliminer une colonne sur deux, ce qui réduit le nombre de pixels par ligne à la moitié.

L'algorithme de [34] est schématisé dans Figure 13 et s'explique comme suit : Soit S_j l'image d'approximation à un niveau de résolution j et soit D_j^X la sous-bande d'orientation X, où $X \in \{\text{H,V,D}\}$, extraite au niveau de résolution j. Comme le montre la figure, l'image S_j passe en entrée de l'algorithme. Elle subit les deux filtrages passe-haut et passe-bas. Les deux images résultantes subissent un sous-échantillonnage sur les lignes. Les deux images sous-échantillonnées sont filtrées chacune par un filtre passe-haut et un filtre passe-bas pour en produire 4 images. Ces dernières sont sous-échantillonnées de nouveau donnant ainsi 4 images de même taille : une image d'approximation (S_{j+1}) et 3 images de détailsD_{j+1}^X ,$X \in \{\text{H,V,D}\}$.

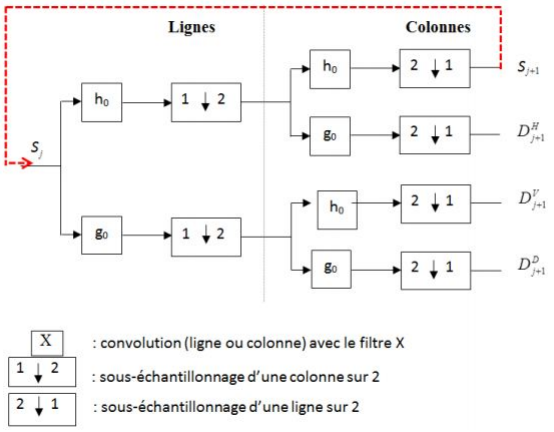
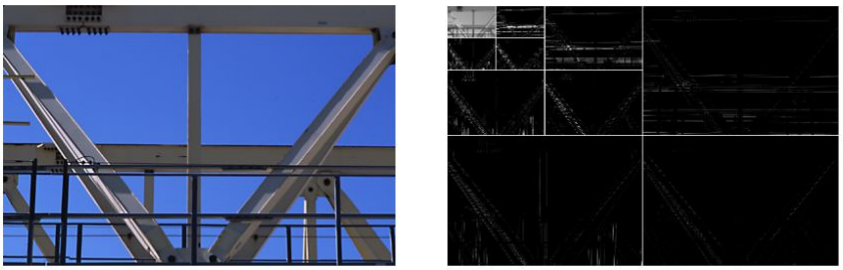


FIGURE 2.1: Schéma de décomposition récursive en ondelettes et l'obtention des quatre sous-bandes.

En plus de bénéficier d'un algorithme de décomposition/reconstruction rapide, la transformée en ondelettes offre plusieurs autres avantages ; la représentation d'image en onde-

lettes est compacte puisque le nombre total des coefficients ondelettes est égal au nombre de pixels de l'image associée. Les coefficients ondelettes sont des coefficients réels contrairement à la transformée de Fourier (FFT) qui fournit des coefficients complexes. Notons aussi que la reconstruction de l'image à partir de sa représentation en ondelettes est parfaite et ceci démontre que toute l'information contenue dans l'image est conservée dans sa transformée en ondelettes. Nous présentons dans Figure 2 un exemple d'application de la transformée en ondelettes à trois niveaux sur une image naturelle.



(a) Image originale (b) Décomposition en ondelettes à 3 niveaux

FIGURE 2.2: Exemple d'application de la transformée en ondelettes à 3 niveaux sur une image naturelle.

2.3 La transformée en contourlettes :

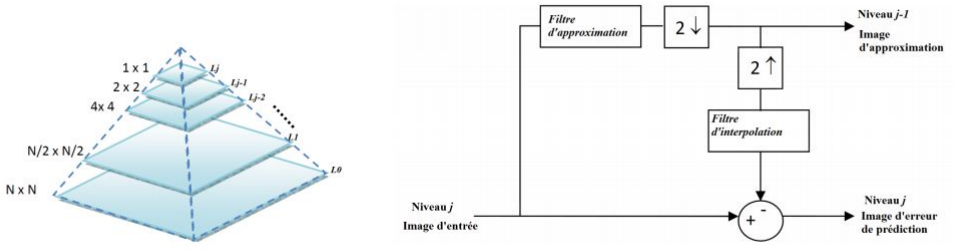
La transformée en contourlettes (CT) a été introduite par Minh N. Do et al [35].En effet, nous expliquons dans les sous-sections 2.3.1 et 2.3.2 comment la transformée en contourlettes dépasse les performances de la transformée en ondelettes. La décomposition en contourlettes comporte deux parties distinctes ; la décomposition en pyramide Laplacienne pour créer les niveaux de résolution de l'image et la décomposition en sous-bandes directionnelles par un banc de filtres bidimensionnels directionnels [36].

2.3.1 La pyramide Laplacienne

La pyramide Laplacienne est introduite en 1983 par Peter J. Burt et Edward H. Adelson [37]. Le but des chercheurs était de trouver une technique permettant de supprimer la forte corrélation entre un pixel et son voisinage. Ceci permet un meilleur codage de l' image puisqu'elle assure l'optimisation du nombre de bits nécessaire pour l'encodage. Cette technique combine les caractéristiques de prédiction et les méthodes de transformation tout en conservant la simplicité et la localité des calculs. La technique de prédiction est relativement simple à implémenter et se considère adaptée aux caractéristiques locales de l'image. La transformée à son tour, permet une meilleure compression de données tout en gardant un temps de calcul raisonnable [37]. Supposons que nous allons calculer la pyramide Laplacienne d'une image $S_0(i, j)$. Pour ce faire, nous appliquons un filtrage passe-bas suivi d'un sous-échantillonnage à cadence 2 selon chaque dimension pour obtenir une image filtrée passe-bas que nous notons$S_1(i, j)$ et dont la taille totale est réduite de 4 par

rapport à l'image originale$S_0(i,j)$. L'image d'erreur de prédiction que nous notons $L_0(i,j)$ est obtenue par l'équation suivante :
$L_0(i,j)$=$S_0(i,j)$ - $S'_1(i,j)$.....(47)

L'image $S'_1(i,j)$ est une version interpolée de $S_1(i,j)$ et ayant donc la même taille que l'image $S_0(i,j)$. À son tour, l'image $S_1(i,j)$ est filtrée par un filtre passe-bas et sous-échantillonnée pour donner une image $S_2(i,j)$. Ainsi, une seconde image d'erreur de prédiction notée $L_1(i,j)$ est calculée en adaptant l'équation (47). En répétant ce processus, nous obtenons une séquence d'images d'erreur de prédiction $L_0(i,j)$,$L_1(i,j)$,$L_2(i,j)$,...,$L_n(i,j)$ tel que la taille de chaque image est le quart de taille de son prédécesseur. Cette séquence est la pyramide Laplacienne de l'image $S_0(i,j)$. Au lieu d'encoder l'image $S_0(i,j)$, il est mieux d'encoder $L_0(i,j)$ et$S_1(i,j)$. En effet, $L_0(i,j)$ est largement dé corrélée et sa représentation binaire nécessite moins de bits que l'image $S_0(i,j)$. De plus, l'image $S_1(i,j)$ est filtrée en passe-bas et peut être alors encodée à un taux d'échantillonnage réduit. Appliquons ce processus plusieurs fois sur les images passe-bas notées $S_k(i,j)$,k∈{ 0, ..., n }, nous bénéficions des 2 avantages cités ci-dessus plusieurs fois de façon récursive. Les valeurs des pixels dans chaque image d'erreur de prédiction dans la pyramide correspondent à la différence entre deux fonctions pseudo-Gaussiennes ou connexes, convoluées avec l'image originale. La différence entre ces deux fonctions est similaire à l'opérateur Laplacien couramment utilisé dans l'amélioration de l'image [38], d'où l'appellation Laplacienne de cette pyramide. Dans Figure 15 nous montrons le fonctionnement de l'algorithme de la pyramide Laplacienne pour produire l'approximation et l'erreur de prédiction (partie (b)) ainsi que le résultat final de cet algorithme (partie (a)).



(a) La pyramide Laplacienne

(b) Système de création de l'approximation et de l'erreur de prédiction

FIGURE 2.3: Schéma de décomposition en pyramide Laplacienne .

2.3.2 Le filtrage bidimensionnel directionnel

La transformée en contourlettes permet un nombre différent et flexible de directions à chaque niveau Laplacien, tout en réalisant l'échantillonnage presque critique. Ceci est assuré par l'utilisation d'un banc de filtres bidimensionnels qui ont différentes orientations. Avec un tel ensemble riche de fonctions de base, la transformée en contourlettes peut représenter un contour lisse avec moins de coefficients par rapport à la transformée en

ondelettes. De plus, la transformée en contourlettes utilise des bancs de filtres itérés, ce qui rend son algorithme rapide et efficace [39]

Pour mieux expliquer ce que nous avons énoncé, nous présentons dans Figure 2.4 un exemple d'application de la transformée en ondelettes et la transformée en contourlettes sur une image, illustrant ainsi le fonctionnement de ces deux transformées. La partie (a) des exemples de cinq images à base d'ondelettes 2-D. La partie (b) montre des exemples de quatre images à base de contourlettes. Dans la partie (c), nous présentons une illustration montrant comment les ondelettes avec des supports carrés ne peuvent pas capturer les points de discontinuités, tandis que les contourlettes ayant des supports allongés peuvent capturer des segments linéaires des contours, et donc peut effectivement représenter un contour lisse avec moins de coefficients [39]

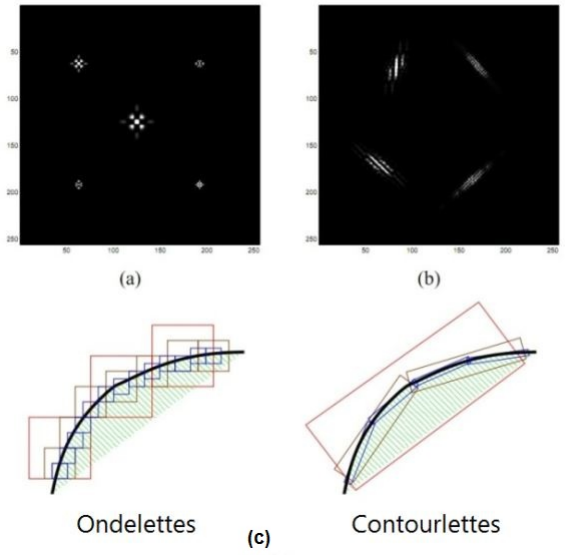


FIGURE 2.4: Représentation du fonctionnement des ondelettes et des contourlettes pour les images

Figure 2.5 montre l'aspect de flexibilité du nombre de directions pour chaque niveau de résolution. En effet, la figure en question est une représentation fréquentielle en contourlettes ayant 4 niveaux de résolutions (L = 4) et les différentes directions fréquentielles pour chaque niveau de résolution ; le premier et le second niveau de résolution ont 4 directions alors que le troisième et le quatrième niveau de résolution ont 8 directions. L'image d'approximation (basses fréquences) ne subit pas de décomposition directionnelle. Nous remarquons aussi que le nombre de directions est toujours une puissance de 2 ($8 = 2^3$ et $4 = 2^2$)

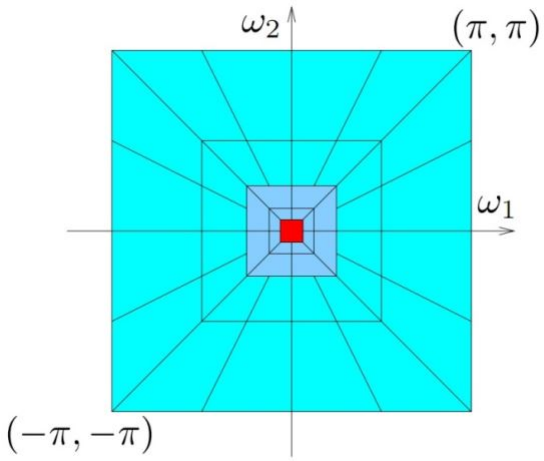


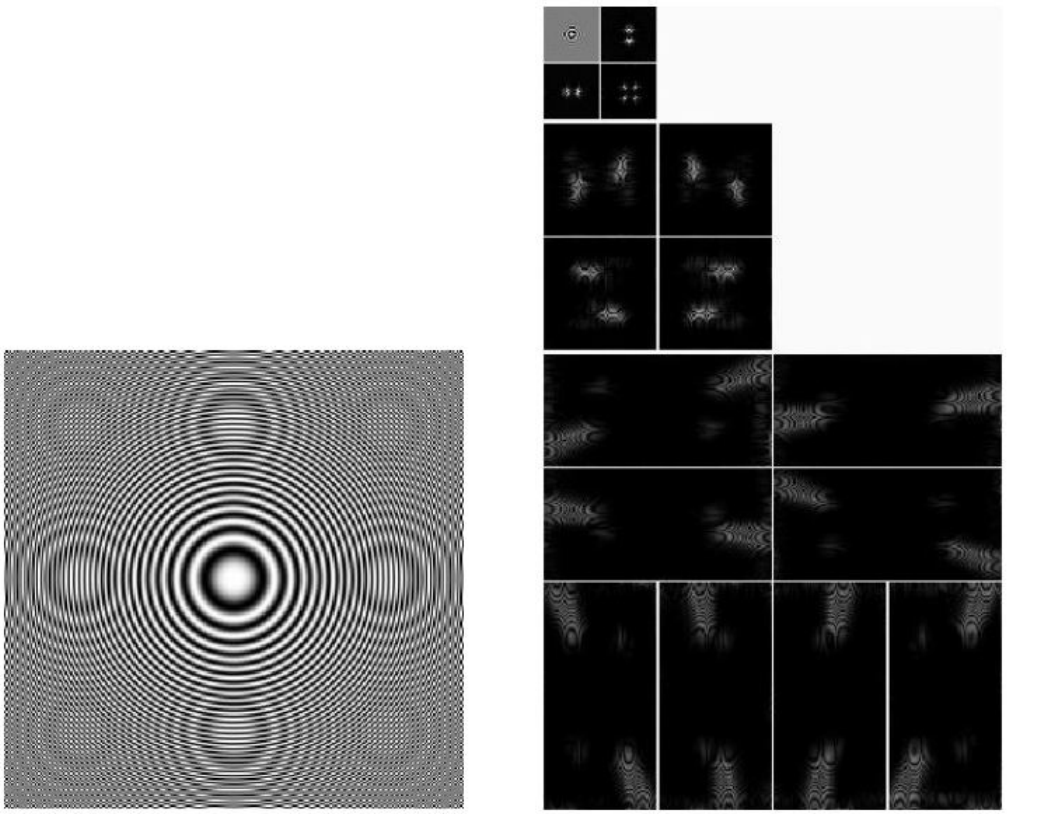
FIGURE 2.5: Représentation fréquentielle de la décomposition en contourlettes ayant 4 niveaux de résolution (L = 4)

Dans Figure 2.6, nous montrons l'exécution de l'algorithme de la transformée en contourlettes sur une image appelée 'zoneplate'. L'image passée comme paramètre d'entrée de l'algorithme de la transformée en contourlettes est représentée par (a) tandis que (b) représente sa transformée en contourlettes. À première vue, il est clair qu'il s'agit d'une transformée en contourlettes à 2 niveaux (L = 2). Nous remarquons aussi que le premier niveau admet 4 directions fréquentielles alors que le deuxième niveau admet 8 directions fréquentielles.
Dans cet exemple, la pyramide Laplacienne a 2 niveaux et 1 approximation. Le banc de filtres contourlettes a subdivisé L_0 en 8 sous-bandes (2 niveaux).L_1 est subdivisée en 4 sous-bandes (1 niveau) et l'image d'approximation est subdivisée en 4 sous-bandes (1 niveau aussi)
Pour illustrer l'aspect de redondance dû à la pyramide Laplacienne nous avons effectué le calcul suivant :

Nombre de pixels de l'image 'zoneplate' :
$256 \times 256 = 65536$ pixels
Nombre de pixels produits par la transformée en contourlettes :
$(32 \times 32) \times 4 + (64 \times 64) \times 4 + (64 \times 128) \times 8 = 86106$ pixels

$$\text{Tauxderedondance} = \frac{nombredepixelsproduitsparlatransformeeencontourlettes}{nombredepixelsdel0imagepasseeenentre} \quad \text{.....(48)}$$

Taux de redondance = 86106/65536 = 1,3138



(a) L'image 'zoneplate' (b) Décomposition en contourlettes à 2 niveaux

FIGURE 2.6: Exemple d'application de la transformée en contourlettes à 2 niveaux

2.4 La transformée en contourlettes redondantes

Suite à une décomposition d'image en contourlettes, il est clair que les bandes résultantes n'ont pas toutes la même taille aussi bien sur un même niveau de résolution qu'à travers les différents niveaux de résolution. Lorsqu'il s'agit d'un traitement mettant en coopération les différentes sous-bandes d'une contourlette, il est souhaitable d'avoir des sous-bandes de la même taille afin d'éviter les problèmes d'interpolation et de mise en correspondance entre les différents niveaux de résolution. À partir de cette exigence, une nouvelle variante de la transformée en contourlettes a vu le jour sous le nom de la transformée en contourlettes redondantes [40]. Nous avons indiqué dans la section (2.3.1) que la redondance provient de la pyramide Laplacienne et peut atteindre un taux de sur-échantillonnage jusqu'à 1/3 donnant un espace total de stockage allant à 4/3 de la taille de l'image d'origine. Plus de redondance peut être obtenue en éliminant toute opération de sous-échantillonnage dans le système pyramidal Laplacien ; en utilisant L filtres passe-bas appropriés pour créer L approximations passe-bas de l'image. La différence entre chaque approximation et sa version ultérieure passe-bas est une bande passante de l'image. Le résultat final est un système de Pyramide Laplacienne Redondante (PLR) avec L+1 niveaux de tailles égales ; une image

d'approximation grossière et L images passe-bande. Chaque image passe-bande est filtrée par un banc de filtres directionnels (BFD). Le filtrage effectué est un filtrage bidimensionnel et les filtres directionnels sont des filtres pseudo-Gaussiens. Le facteur de redondance est défini par L. En appliquant le même principe sur le niveau de décomposition d-directionnel avec échantillonnage critique sur chaque bande passante PLR, nous obtenons une transformée en contourlettes redondantes avec L D sous-bandes directionnelles de tailles égales, en plus de l'image d'approximation grossière. Le symbole D représente le nombre de directions souhaité, et chaque sous-bande contourlettes est représentée par une sous-image. Pour mieux comprendre le fonctionnement de la décomposition en contourlettes redondante, nous exposons dans Figure 2.7 le schéma de son algorithme.

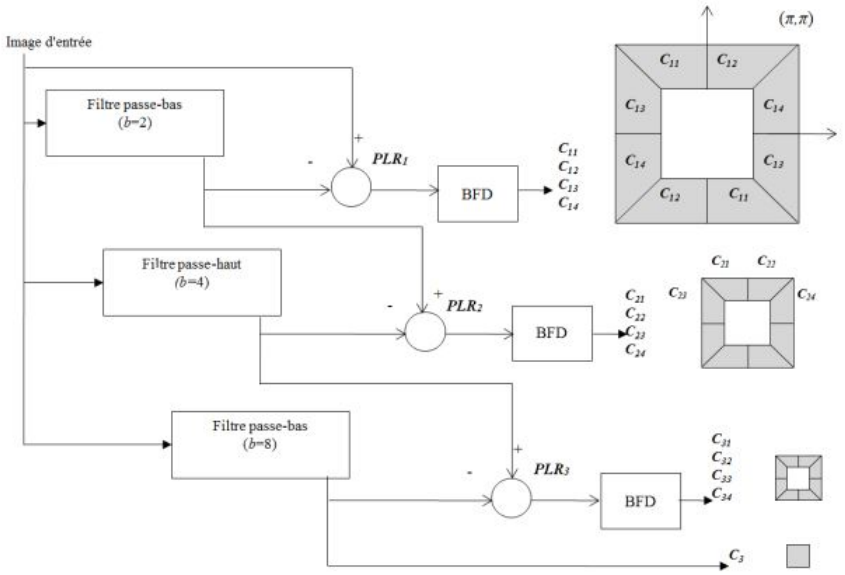
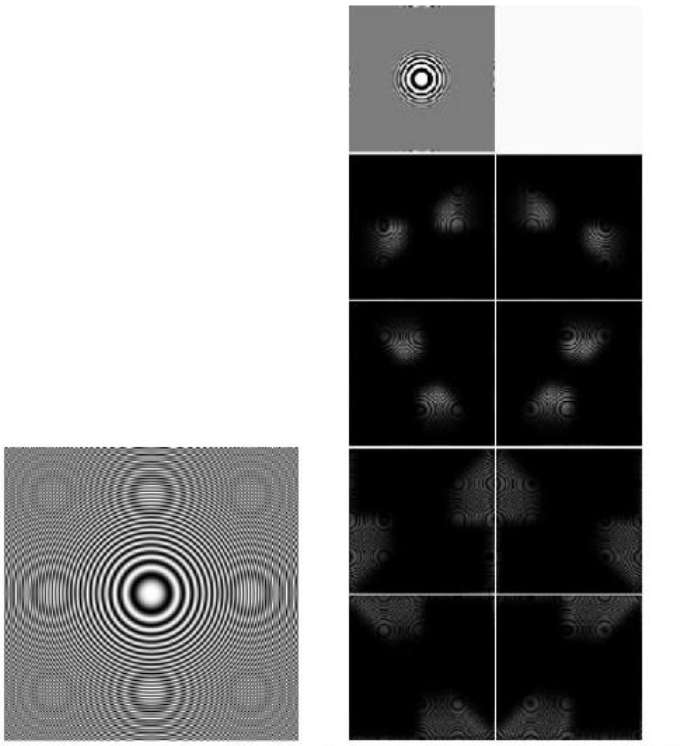


FIGURE 2.7: Schéma de décomposition en contourlettes redondantes.

Dans Figure 2.8, nous présentons l'application de la transformée en contourlettes redondantes sur la même image de Figure 2.6. La partie (a) représente l'image passée en entrée de l'algorithme de la transformée en contourlettes redondantes, tandis que la partie(b) représente les sous-bandes de la transformée.



(a) L'image 'zoneplate' (b) Décomposition en contourlettes redondantes

FIGURE 2.8: Exemple d'application de la transformée en contourlettes redondantes.

2.5 La distribution gaussienne généralisée :

2.5.1 définition :

La distribution gaussienne est un model typique pour les signaux et le bruit dans de nombreuses applications en sciences et en génie. Cependant, il existe certaines applications où cette hypothèse gaussienne s'écarte de comportement réel aléatoire . la distribution généralisée de Gauss a été proposé pour la modélisation du bruit atmosphérique, sous-bande de codage audio et signaux de vidéo [41], bruit impulsif, direction d'arrivée, indépendantes d'analyse des composants [42], séparation des signaux aveugle[43], GARCH [44], etc...

La distribution gaussienne généralisée (GG) peut être paramétrée de telle manière que sa moyenne (μ) et sa variance (σ^2) coïncident avec la distribution gaussienne. le GG a le paramètre de forme p, qui est une mesure du gibbosité(peakedness) de la distribution, cependant, il semble qu'il n'existe pas une expression proche pour estimer p. Le paramètre p détermine la forme de la distribution. ex., la distribution gaussienne est obtenu pour (p = 2), la distribution laplacienne pour (p = 1), et en faisant p=0 nous pouvons obtenir une distribution proche de la distribution uniforme. Dans la plupart des applications, la

moyenne peut être considéré comme zéro, alors nous nous concentrerons sur l'estimation du paramètre de forme du Distribution GG avec deux paramètres, soit$\mu = 0$.

Varanasi et Aazhang [45] discutent de l'estimation des paramètres pour la GG en utilisant les méthodes Rodriguez-Dagnino et León-Garcia [46] présentent estimateur à formulaire proche basé sur l'inégalité de Gurland. Il y a quelques difficultés de calculs concernant les expressions mathématiques présentées dans [46], principalement liées à la fonction gamma, ce qui est important pour l'encodage de sous-bande des signaux vidéo, et certains applications connexes. Cependant, l'intervalle n'est pas assez large pour couvrir la plupart des cas.

2.5.2 principe de GGD :

Une variable aléatoire X est distribuée en gaussienne généralisée si sa fonction de densité de probabilité (pdf) est donné par :

$$gg(x; \mu, \sigma, p) = \frac{1}{2\Gamma(1 + 1/p)A(p, \sigma)} e^{-|\frac{x-\mu}{A(p,\sigma)}|^p}, x \in R.....(49)$$

où

$$\mu \in R, p, \sigma > 0; A(p, \sigma) = \left[\frac{\sigma^2\Gamma(1/p)}{\Gamma(3/p)}\right]^{1/2} .....(50)$$

Le paramètre μ est la moyenne, la fonction $A(p, \sigma)$ est un facteur d'échelle qui permet que $Var(X) = \sigma^2$, et p est le paramètre de forme. Comme nous le constatons ci-dessus. p = 2 correspond à une distribution gaussienne, alors que dans les cas limitatifs $p \to +\infty$ le pdf de l'équation (47) converge vers une distribution uniforme en $(\mu - \sqrt{3}\sigma, \mu + \sqrt{3}\sigma)$, et lorsque $p \to 0+$ la distribution devient une distribution dégénérée en $X \sim GG(\mu, \sigma, p)$ pour indiquer que X est un variable aléatoire avec pdf comme dans l'équation (47), et nous indiquerons $GG(\sigma, p) = GG(0, \sigma, p)$.

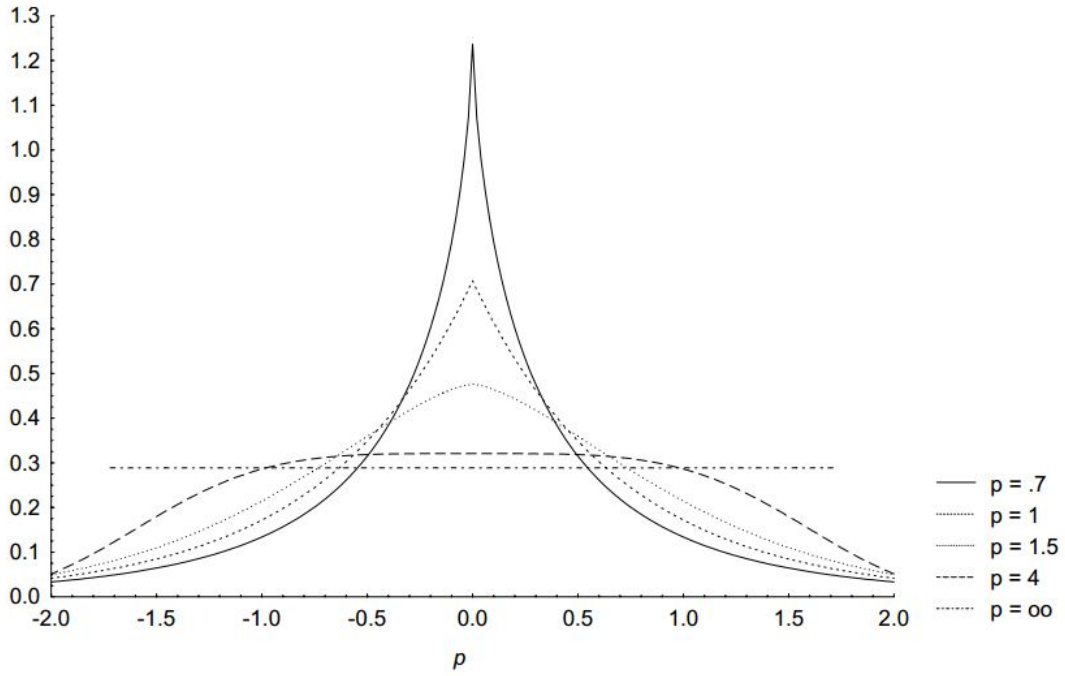


FIGURE 2.9: Généralisé PDF gaussiens pour différentes valeurs de p. Du haut vers le bas : p = 0.7, 1, 1.5, 2, 4 .

la distribution GG est symétrique par rapport à μ , de ce fait les moments centraux impairs sont nuls, c à d $E(X-\mu)^r = 0, r = 1, 3, 5, ...$ les moments centraux pairs peuvent être obtenu de l'absolu des moments centraux donnés par :

$$E|X-\mu|^r = \left[\frac{\cdot\sigma^2\Gamma(1/p)}{\Gamma(3/p)}\right]^{r/2}\frac{\Gamma(\frac{r+1}{p})}{\Gamma(1/p)} \text{.....(51)}$$

En particulier, la variance de X est :

$$Var(X) = E(X-EX)^2 = E(X-\mu)^2 = EY^2 = \sigma^2 \text{.....(52)}$$

2.6 distance de Kullback-Leibler :

2.6.1 Définition :

La divergence KL est une mesure de la similitude (ou de la différence) des deux distributions de probabilité .Pour mesurer la différence entre deux distributions de probabilité sur le même variable x, une mesure, appelée la divergence Kullback-Leibler, ou simplement, le KL de divergence, a été largement utilisé dans la littérature d'exploration de données. Le concept a été créé dans la théorie des probabilités et la théorie de l'information.

2.6.2 mesure pour distance de Kullback-Leibler :

Soit deux distribution discrète de probabilité p et q, la divergence KL pour un ensemble de points i est définie comme suit :

$$D_{KL}(p||q) = \sum_i p_i log_2 \frac{p_i}{q_i} .....(53)$$

D_k est non négatif ($\rangle = 0$), pas symétrique en p et q, zéro si les distributions correspondent exactement et peuvent potentiellement être égales a l'infinie. Une interprétation technique courante – bien que dépourvue d'intuition – est que la divergence KL est associée au choix d'une distribution q pour se rapprocher de la vraie distribution p[47]. Une compréhension intuitive, cependant, découle de la théorie de la probabilité - la probabilité que l'on observe un ensemble de données étant donné qu'un modèle particulier était vrai [48]. Prétendre que nous effectuons une expérience pour mesurer un discret, variable aléatoire - comme lancer un dé plusieurs fois . un histogramme $c = \{c_i\}$, $n = \sum_i c_i$. Cet histogramme mesure la fréquence relative de chaque face de la matrice (ou, chaque type de motif de tir). Si cette expérience dure éternellement, les comptes histogrammes normalisés $\frac{c_i}{n}$ reflètent une distribution sous-jacente $p_i = \frac{c_i}{n}$ Pour obtenir une certaine intuition, imaginez que nous avons effectué n = 1 mesures - dans ce cas, la probabilité serait le q_i à chaque fois attribué au motif de tir unique observé. La probabilité L rétrécit mutiplicativement que nous effectuons plus de mesures (ou n pousse). Idéalement, nous voulons que la probabilité soit invariante au nombre de mesures - ce est donné par la probabilité moyenne $\bar{L} = L^{\frac{1}{n}}$,un nombre entre 0 et 1. Intuition correspondant, que nous effectuons plus mesures,si $\frac{c_i}{n} \to q_i$, alors la probabilité moyenne serait parfaite, ou $\bar{L} \to 1$. Inversement, comme $\frac{c_i}{n}$ diverges du modèle q_i, la probabilité moyenne $\bar{L}$ diminue, approchant zéro. Le lien entre la probabilité et la divergence de KL découle du fait que si nous effectuons un nombre infini de mesures [38]

$$D_{kl}(p||q) = -log_2 \bar{L} .....(54)$$

Ainsi, si les distributions p et q sont identiques,$\bar{L} = 1$ et $D_{KL} = 0$. L'intuition centrale est que la divergence KL mesure effectivement la probabilité moyenne d'observer des données (infinies) avec la distribution p si le modèle q a effectivement généré les données.

La divergence KL a de nombreuses applications et est un fondement de la théorie de l'information et des statistiques [47]. Par exemple, on peut se demander dans quelle mesure une distribution conjointe p(x, y) est similaire au produit de ses marginaux p(x)p(y) - c'est l'information mutuelle, une mesure générale de la dépendance statistique entre deux variables aléatoires [47],

$$I(X;Y) = \sum_{x,y} p(x,y) log_2 \frac{p(x,y)}{p(x)p(y)} .....(55)$$

L'information mutuelle est nulle si et seulement si les deux variables aléatoires X et Y sont statistiquement indépendantes. En plus de son rôle dans l'information mutuelle, la

divergence KL a été largement appliquée dans le codage neural , plus récemment pour quantifier les effets de la dépendance conditionnelle entre les neurones [49] et pour mesurer dans quelle mesure les corrélations d'ordre supérieur peuvent être approché par la structure d'ordre inférieur [50].

2.7 CONCLUSION

Dans ce chapitre, nous avons étudié trois types de transformées multirésolution : les ondelettes, les contourlettes et les contourlettes redondantes. Les ondelettes, bien qu'elles soient toujours un outil puissant et largement utilisé dans le traitement et l'analyse d'images, elles admettent des limites relatives à la directionnalité. Les contourlettes viennent dans le but d'améliorer le comportement des ondelettes. Par conséquent, les contourlettes fournissent un bon support pour l'analyse multirésolution et ceci grâce à la flexibilité de choisir la directionnalité des sous-bandes. Ainsi, la détection des contours lisses est meilleure. Nous avons aussi présenté dans ce chapitre la transformée en contourlettes redondantes qui est une variante de la transformée en contourlettes permettant d'avoir des sous-bandes de la même taille, facilitant ainsi le traitement hiérarchique de l'image.

Le chapitre présente également une façon pour la représentation d'une distribution de probabilité correspondant à une variable aléatoire , la GGD permet grâce à son paramètre de forme de couvrir une lange palette de probabilité .Nous avons vu également la KLD qui permet de mesurer la divergence (similitude) entre deux distribution de probabilité .

CHAPITRE 3

CHAPITRE 3

3.1 Introduction

Après avoir étudié le domaine de Contourlet Redandante , GGD et KLD.L'implémentation d'une application d'un système de recherche d'images devient une nécessité afin d'avoir une vue plus claire. Dans ce chapitre nous allons exposer les différentes étapes par lesquelles nous sommes passés pour la réalisation de notre application.

3.2 Méthode utilise

Dans ce chapitre, nous parlons de la manière dont nous pouvons implémenter les règles que nous avons étudiées dans les chapitres précédents. Nous en avons expliqué l'aspect théorique, comme nous avons utilisé certaines des bibliothèques de python.

3.2.1 Architecture De Logiciel

Pour simplifier les choses, nous avons préparé une architecture qui résume les étapes que nous avons effectuées et les méthodes que nous avons utilisées lors de notre mise en œuvre du logiciel voir la figure (3.1).

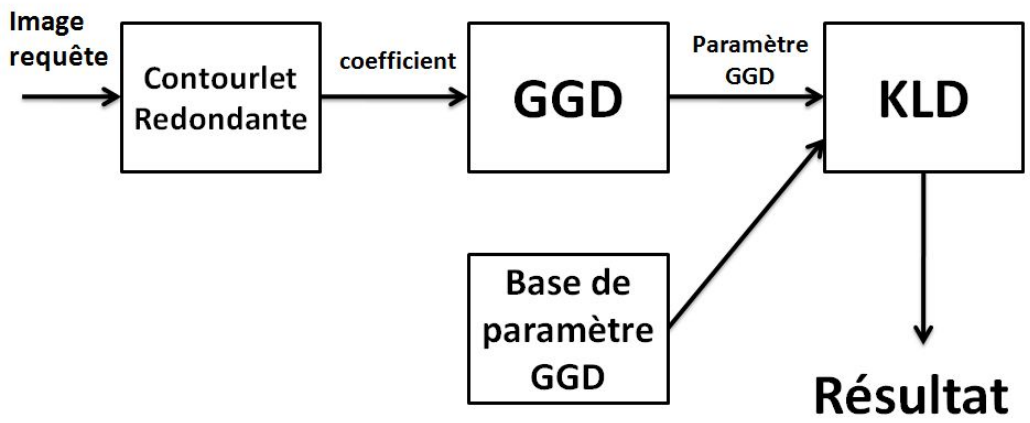


FIGURE 3.1: Architecture de les étapes qui passer au logiciel.

Nous préparons la base de données de paramètres GGD dans figure (3.1) une fois pour augmenter la vitesse, comme suit :

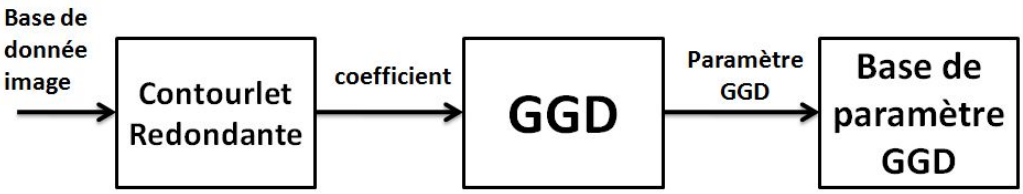


FIGURE 3.2: Architecture de les étapes qui passer au Base de donnée.

3.2.2 Contourlet Redondante

Nous avons vu précédemment l'aspect théorique du contourlet (il se compose de pyramide laplacienne et bank de filtre) qui donne des coefficient et qui à son tour nous donne des résultats acceptables lors de la recherche.Pour clarifier davantage la question sur les résultats pour le contourlet, nous fournissons un exemple simple

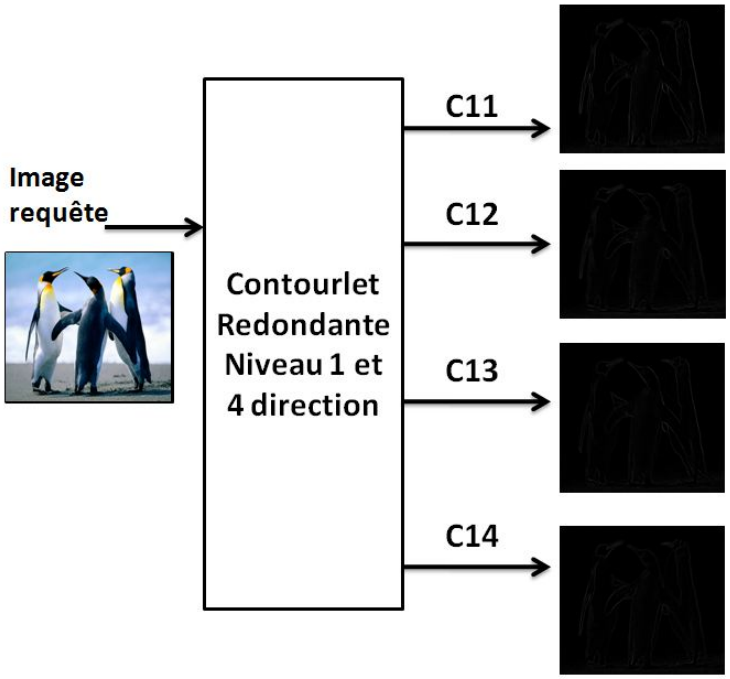


FIGURE 3.3: un exemple simple de résultats du contourlet par niveau 1 et 4 direction.

3.2.2.1 Mise en œuvre de la pyramide laplacienne

Nous parlons maintenant de la première phase de contourlet redondante représentée par la pyramide laplacienne et nous avons pour objectif de supprimer la forte corrélation entre

un pixel et son voisinage. Le côté théorique à été introduit dans le chapitre précédent , et maintenant nous parlons du côté pratique.
Pour appliquer cela, nous avons utilisé les relations mathématiques suivantes :

G_0 = Image originale

L = 3

$G_i(f1, f2) = G_{i-1}(f1, f2).H_i(f1, f2)$.....(56) , i=1,2...,L

$LP_i(m, n) = G_{i-1}(m, n) - G_i(m, n)$, ...(57)i=1,2. . . ,L

$LP_{L+1}(m, n) = G_L(m, n)$.....(58)

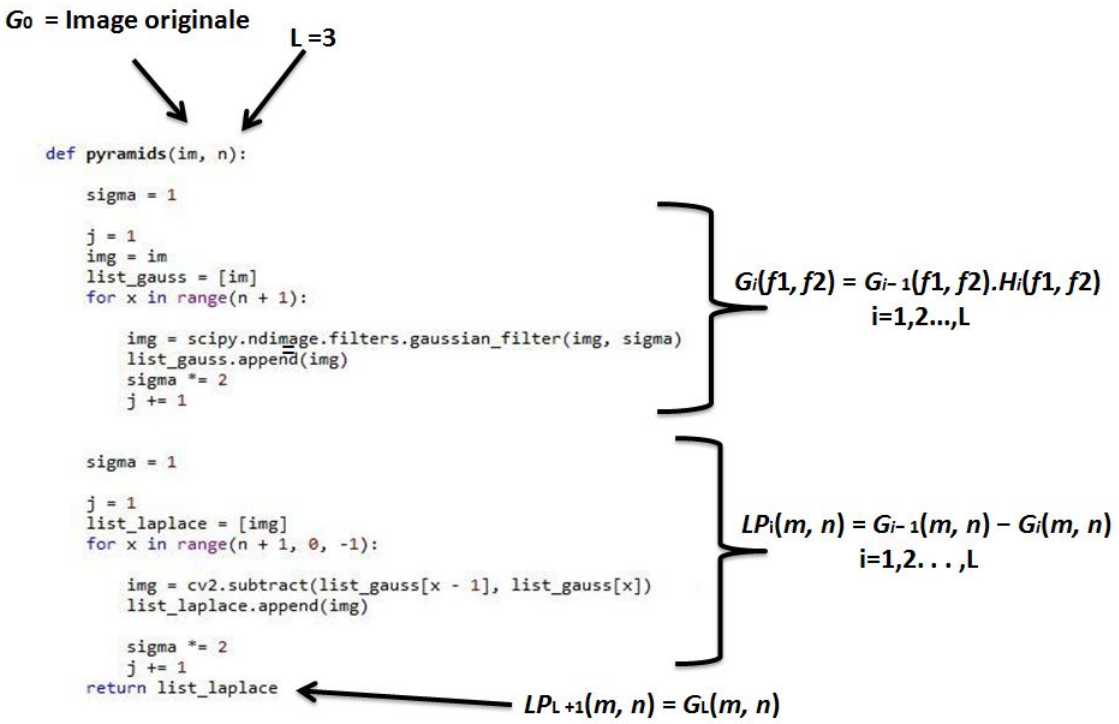


FIGURE 3.4: Représentation de code on python de la fonction qui faire la pyramide laplacienne.

on a utiliser des package de la bibliothèque scipy pour implémenter la méthode gaussienne et package de cv2 pour faire la substraction entre gaussienne de deux images différentes .

3.2.3 Bank de filtre

L'application du bank de filtre ,génère un coefficient pour chaque niveau et direction .Nous avons besoin de caractériser les coefficient générer , chose qui est fait grâce a la GGD , le calcule du paramètre GGD est expliquer dans ce qui suit .
Pour l'implémentation nous avons utilisé la bibliothèque **scipy.signal** utilise :

```
def filtre_bank (h2):

    img1 = scipy.signal.convolve(h2,hhor,mode='same')

    img2 = scipy.signal.convolve(h2,hver,mode='same')

    img3 = scipy.signal.convolve(h2,hd,mode='same')

    img4 = scipy.signal.convolve(h2,hg,mode='same')

    height, width = h2.shape[:2]
    c=int(height/2)
    d=int(width/2)

    return img1,img2,img3,img4
```

FIGURE 3.5: Représentation de code on python de la fonction qui faire le filtre bank.

La figure suivante montre un exemple de décomposition de l'image Lena (512x512) en Contourlet redondante.

FIGURE 3.6: Représentation fréquentielle d'une décomposition en Contourlet redondante avec 3 niveaux de résolution (L=2) et 4 directions fréquentielles (D=4). L'approximation passe-bas de l'image (C_3) n'est pas décomposée par le banc de filtres directionnels

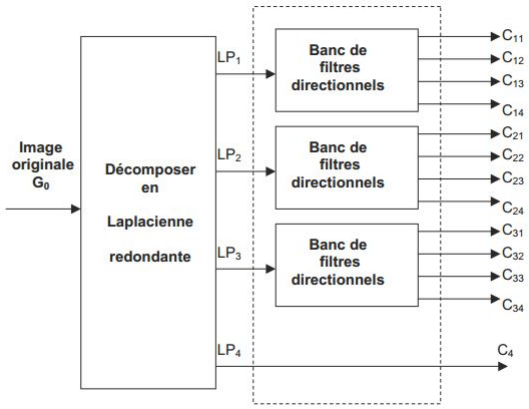


FIGURE 3.7: Diagramme de décomposition en Contourlet redondante à 4 niveaux de résolution (L=3) et à 4 directions fréquentielles (D=4). La décomposition en Laplacienne redondante est détaillée dans la Figure (3.4) alors que le bank de filtres directionnels utilisé est détaillé dans le Figure (3.5). L'image originale de taille MN est ainsi transformée en 12 sous-images de tailles MN et une approximation basse-fréquence de taille MN (C_4).

3.3 Distribution de Gaussien Généralisée

GGD a deux paramètres : le paramètre γ contrôle la « forme » de la distribution, c'est-à-dire la vitesse d'atténuation ; σ contrôle la variance.

La caractérisation de coefficient contourlet est effectuée en les représentant sous forme d'une GGD , les deux paramètre qui définissent la GGD vont être calculés est stockées pour les utiliser dans notre système d'indexation , La distribution gaussienne généralisée est définie dans le chapitre précédent (les formules 49 et 50) comme suit :

$$f(x;\alpha,\sigma^2) = \frac{\alpha}{2\beta\Gamma(1/\alpha)} \exp\left(-\left(\frac{|x|}{\beta}\right)^{\alpha}\right)$$

where

$$\beta = \sigma\sqrt{\frac{\Gamma(1/\alpha)}{\Gamma(3/\alpha)}}$$

and $\Gamma(\cdot)$ is the gamma function:

$$\Gamma(a) = \int_0^{\infty} t^{a-1}e^{-t}dt \quad a > 0.$$

FIGURE 3.8: La distribution gaussienne généralisée .

on a la fonction suivant de la méthode GGD qui donnée la vecteur et le résultat sigma et gamma paramètre .

```
def estimate_GGD_parameters(vec):
    gam =np.arange(0.2,10.0,0.001)
    r_gam = (gamma(1/gam)*gamma(3/gam))/((gamma(2/gam))**2)
    sigma_sq=np.mean((vec)**2)
    sigma=np.sqrt(sigma_sq)
    E=np.mean(np.abs(vec))
    r=sigma_sq/(E**2)
    diff=np.abs(r-r_gam)
    gamma_param=gam[np.argmin(diff, axis=0)]
    return  gamma_param,sigma
```

tel que [0,2: 0,001: 10], c'est-à-dire que α est compris entre (0,2, 10) tous les 0,001.

Calcule les paramètre GGD

FIGURE 3.9: Représentation de code on python de la fonction qui faire GGD .

Résultat de deux image dégrader

Afin de montrer la précision de GGD , prenons l'exemple de ces deux images qui sont presque identique visuellement (l'une et une simple dégradation de l'autre) l'histogrammes des coefficient contourlet sont proches et la GGD permet de distinguer la similitude .

FIGURE 3.10: deux image dégrader de test.

FIGURE 3.11: Décomposition des deux images dégrader Monshot (512x512) en représenter la coefficient C11 de chaque image.

Maintenant on a représenter l'histogramme de la coefficient précédant .

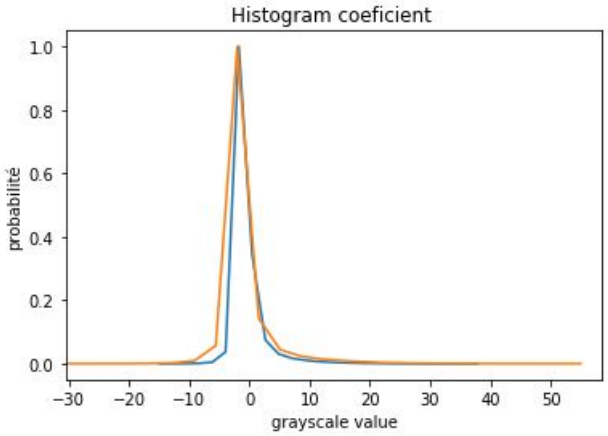


FIGURE 3.12: histogramme de la résultat de la figure (3.11).

couleur bleu représenter histogramme de l'image Monshot et la couleur rouge représenter l'histogramme de l'image dégrader pour Monshot .

l'application GGD a les deux histogramme (figure 3.8) et donnée le résultat suivante de même coefficient :
- γ_{blue} =0.34200000000000014, σ_{blue} =63.573276141474665 .
- γ_{rouge} =0.35200000000000015, σ_{rouge} =16.512313739799392 .

Résultat de deux image différente

Lorsque deux images différentes (Monshot et Lena) sont utilisées, le paramètre GGD est significativement différent .

FIGURE 3.13: les deux image différentes utilise dans le teste (Monshot et Lena.

Maintenant on a représenter la coefficient de chaque image déférente pour Monshot et Lena (figure 3.9).

FIGURE 3.14: Décomposition des deux images différente Monshot et Lena (512x512) en représenter la coefficient C11 de chaque image.

Maintenant on a représenter l'histogramme de la coefficient précédant .

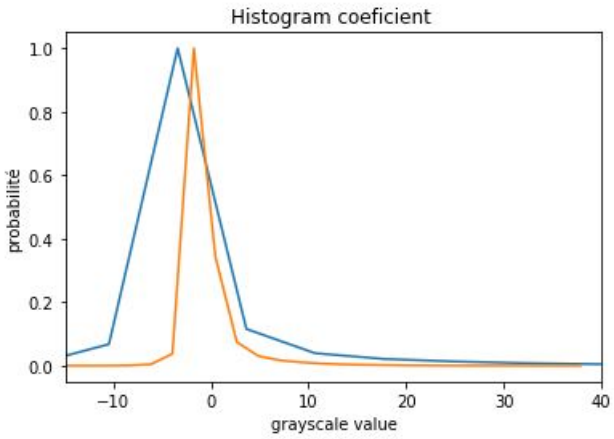


FIGURE 3.15: histogramme de la résultat de la figure (3.14).

couleur rouge représenter histogramme de l'image Monshot et la couleur bleu représenter l'histogramme de l'image Lena .

l'application GGD a les deux histogramme (figure 3.11) et donnée le résultat suivante de même coefficient :
- γ_{blue} =0.34200000000000014, σ_{blue} =63.573276141474665 .
- γ_{rouge} =0.35500000000000015, σ_{rouge} =6.689220594639407 .

3.3.1 Kullback–Leibler divergence.

Nous en avons déjà parlé dans le chapitre précédant , nous donnons maintenant une partie de son code pour préciser qu'ils ont ajouté à certains des résultats.avant tout sa on représente l'algorithme de cette méthode :

Inputs: $S_{t-1} = \{\langle x_{t-1}^{(i)}, w_{t-1}^{(i)}\rangle \mid i = 1, \ldots, n\}$ representing belief $Bel(x_{t-1})$, control measurement u_{t-1}, observation z_t, bounds ε and δ, bin size Δ

$S_t := \emptyset$, $n = 0$, $k = 0$, $\alpha = 0$ /* *Initialize* */

do /* *Generate samples . . .* */

Sample an index $j(n)$ from the discrete distribution given by the weights in S_{t-1}

Sample $x_t^{(n)}$ from $p(x_t \mid x_{t-1}, u_{t-1})$ using $x_{t-1}^{(j(n))}$ and u_{t-1}

$w_t^{(n)} := p(z_t \mid x_t^{(n)})$; /* *Compute importance weight* */

$\alpha := \alpha + w_t^{(n)}$ /* *Update normalization factor* */

$S_t := S_t \cup \{\langle x_t^{(n)}, w_t^{(n)}\rangle\}$ /* *Insert sample into sample set* */

if ($x_t^{(n)}$ falls into empty bin b) **then** /* *Update number of bins with support* */

$k := k + 1$

$b :=$ non-empty

$n := n + 1$ /* *Update number of generated samples* */

while ($n < \frac{1}{2\varepsilon}\chi^2_{k-1,1-\delta}$) /* *. . . until K-L bound is reached* */

for $i := 1, \ldots, n$ **do** /* *Normalize importance weights* */

$w_t^{(i)} := w_t^{(i)}/\alpha$

return S_t

FIGURE 3.16: Algorithme de KLD.

Dans le Logiciel on appelle la fonction kldivergence(p, q) qui faire la distance entre les résultat de GGD (gamma paramètre et sigma paramètre .)

```
def kldivergence(p, q) :

    return sum(p[i] * log2(p[i]/q[i]) for i in range(len(p)))
```

Les résultats de l'application avec la même image sont les suivants :

FIGURE 3.17: Décomposition d'une image (512x512) en GGD à L=3 et D=4. Les coefficients Contourlet ci-dessus , en utilise la distance de KLD Pour les deux mêmes images .

- kld pour les mêmes images = 0.0 pour la coefficient de la niveau 1.
- kld pour les mêmes images = 0.0 pour la coefficient de la niveau 2.
- kld pour les mêmes images = 0.0 pour la coefficient de la niveau 3.
- kld pour les mêmes images = 0.0 pour la coefficient de la niveau 4.

les résultats KLD pour les différentes images (Monshot et Lena) en chaque niveau (figure 3.13) :

- kld des différentes images = 4.700312690454774 pour la coefficient de la niveau 1.
- kld des différentes images = 3.863398679245438 pour la coefficient de la niveau 2.
- kld des différentes images = 4.079874920536208 pour la coefficient de la niveau 3.
- kld des différentes images = 4.720979991519425 pour la coefficient de la niveau 4.

3.4 Simulation et Teste

3.4.1 Base d'images utilisées

Pour évaluer notre méthode et tester ses performances selon différents angles,nous avons choisi d'utiliser la base d'images COREL. Au total, il y a 10,800 images d'usage général dans la base de données partitionnées en 80 groupes de concepts, par exemple : automne, aviation, bonsaï, château, nuage, chien, éléphant, iceberg, primates, navire, stalactite, moteur à vapeur, tigre, train et chute d'eau, comme le montre la figure suivante. La taille des chaque images est soit 80120 soit 12080.

La figure suivante montre un exemple des images présentes dans la base utilisée.

FIGURE 3.18: exemple des images présentes dans la base utilisée .

3.4.2 Résultat Obtenu

Pour tester l'efficacité de notre méthode sur la base d'image choisi certain mesures trouver sont calculer afin de refléter la performance du système implémentés , ces taux sont : vrais positive,vrais négative , faux positive et faux négative.

Un vrai positif (TP) est un résultat où le modèle prédit correctement la classe positive. De même, un vrai (TN) négatif est un résultat où le modèle prédit correctement la classe négative.Un faux positif (FP) est un résultat où le modèle prédit incorrectement la classe positive. Et un faux négatif (FN) est un résultat où le modèle prédit incorrectement la classe négative.

VP= requête prédit positive/ requête totale .

VN= requête prédit négative/ requête totale .

FP= requête non prédit positive/ requête totale .

FN= requête non prédit négative/ requête totale . Les résultats étaient les suivants :

seille	$N°$Niveau	FN	TN	FP	TP	Pourcentage
0.001	4	2.0 %	98.0%	0.0 %	100.00%	90.20 %
0.17	4	3.085%	96.915%	0.11 %	99.89%	90.21 %
0.23	4	4.577%	95.423%	0.36 %	99.64%	90.14 %
0.001	3	2.0 %	98.0 %	0.0 %	100.00%	90.20 %
0.17	3	3.198 %	96.802 %	0.136%	99.86 %	90.20%
0.23	3	4.682 %	95.318%	0.399 %	99.601%	90.11 %
0.001	2	2.0%	98.0%	0.0 %	100.0%	90.2 %
0.17	2	5.736%	94.264%	1.199 %	98.801%	89.495 %
0.23	2	8.044%	91.956%	2.222 %	97.778%	88.805 %

3.5 Avantage

Notre programme présente de nombreux avantages qui sont :

1- fiabilité : performant sur l'image de différentes caractéristique visuelle .

2 - Vitesse de recherche malgré la grande base de données (le tempe d'exécution est 0.75 seconde et pour préparer la base de donnée la premiere fois en attente 5 minute).

3 -Permet de choisi le seuil et le niveau appliquée et malgré le changement des ces deux paramètre , les résultats sont toujours trés acceptable .

4- Le pourcentage d'images dont il n'a pas besoin et qu'il ne retire pas (TN) est grand et en tout (niveau) et seuil n'est jamais en dessous de 90%

3.6 Critique

Notre programme présente de nombreux avantages, mais il y a certains problèmes que nous aimerions améliorer à l'avenir :

1- Représentation GGD des coefficient n'est pas parfait on peut prévoir a l'avenir l'utilisation MOGGD qui permet une représentation presque parfait des coefficient .

2- La différence visuelle n'est pas visible seulement au niveau de la luminance (niveau de gris), elle peut plus distinguée au niveau des couleur , dans ce cas il est impérativement demandé d'utiliser la version Multi-variée de la GGD.

3- Le KLD n'est qui une forme approximative de la distance (déjà elle n'est pas symétrique), on peut améliorer un autre façon plus performante pour calculer la distance .

3.7 CONCLUSION

Ce chapitre à été conçu pour expliquer l'implémentation et la simulation des outils théorique vue précédemment .Le système de recherche d'image proposé et construit autour dz ces outils (Contourlet Redandante , GGD et KLD) a montrer son efficacité et haute performance . On peut dire que ce système déposée plein d'autre proposés dans la littérature , malgré qu'on n'a pas eu le temps pour faire une étude comparative.