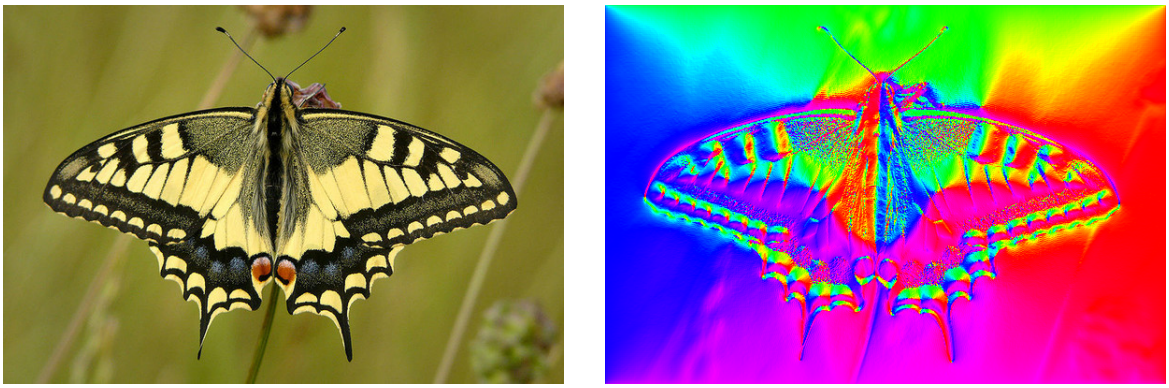
Transformée de Riesz pour le calcul de l'orientation locale dans les images et vidéos

Compte-rendu de projet

NGANKAM Franck
SEPPECHER Coline

La transformée de Riesz est très utilisée en traitement du signal, par exemple pour la démodulation de franges d'interférence ou d'hologrammes. Elle permet de déterminer les caractéristiques locales d'une image. Ces dernières années, un groupe de chercheurs du MIT [1] a introduit une méthode d'amplification de mouvement dans les vidéos basée sur cet outil. Grace à une analyse de la phase du signal, ils ont développé un microscope à amplification de mouvement afin de rendre visible des mouvements imperceptibles à l'oeil nu telle que le rythme respiratoire, le flux sanguin, ou encore la fréquence cardiaque.

L'objectif de ce projet est d'étudier le fonctionnement de la transformée de Riesz et de proposer un algorithme de calcul efficace afin de rendre possible les applications à la vidéo.



(a) Le Machaon ou Grand porte-queue (b) Orientation locale

Figure 1: Utilisation de la transformée de Riesz pour déterminer l'orientation locale d'une image

Plan du compte-rendu

1 Transformée de Riesz

1.1 Transformée de Hilbert et signal analytique

Pour commencer, on se place en dimension 1. En traitement du signal, on associe à un signal réel $x(t)$ un signal analytique $x_A(t)$ pour lequel les composantes de fréquences négatives sont supprimées. $x_A(t)$ est un signal complexe dont la partie réelle s'identifie à $x(t)$ et dont la partie imaginaire est la transformée de Hilbert de $x(t)$ notée $H_x(t)$:

$$x_A(t) = x(t) + iH_x(t)$$

La représentation analytique d'un signal permet de définir son amplitude A et sa phase Φ en le mettant sous forme exponentielle :

$$x_A(t) = Ae^{i\Phi(t)} \text{ où } A = |x_A(t)|$$

La transformée de Hilbert d'une fonction réelle f peut s'écrire très simplement à l'aide de la transformée de Fourier :

$$\hat{H}_f(\nu) = -i\frac{\nu}{|\nu|}\hat{f}(\nu)$$

où $\hat{f}$ est la transformée de Fourier de f et $\hat{H}_f$ la transformée de Fourier de H_f.

1.2 Transformée de Riesz et signal monogène

Dans notre étude, nous traitons des images c'est à dire des fonctions en dimension 2. La transformée de Riesz (abrégée $\mathcal{TR}$ par la suite) permet la généralisation isotropique de la transformée de Hilbert en dimension 2 (et plus généralement en dimension n). Soit $f \in L^2(\mathbb{R}^2)$ une fonction réelle. Sa transformée de Riesz, notée $R_f = (R_{f,1}, R_{f,2})$, est la fonction vectorielle définie dans le domaine de Fourier par :

$$\forall \vec{\xi} \in \mathbb{R}^2, \hat{R}_f(\vec{\xi}) = -i\frac{\vec{\xi}}{\|\vec{\xi}\|_2}\hat{f}(\vec{\xi}) \tag{1.1}$$

Le signal monogène de f, généralisation du signal analytique en dimension 2, s'écrit alors dans un repère cartésien $(\vec{i}, \vec{j}, \vec{k})$:

$$f_M(\vec{x}) = R_{f,1}(\vec{x})\vec{i} + R_{f,2}(\vec{x})\vec{j} + f(\vec{x})\vec{k}$$

Le passage en coordonnées sphériques (*figure 2*) permet de récupérer les caractéristiques locales de l'image f : amplitude locale A, phase locale Φ et orientation locale θ.

$$\begin{cases} R_{f,1} &= A\sin(\Phi)\cos(\theta) \\ R_{f,2} &= A\sin(\Phi)\sin(\theta) \\ f &= A\cos(\Phi) \end{cases} \qquad \begin{cases} A &= \sqrt{R_{f,1}^2 + R_{f,2}^2 + f^2} \\ \Phi &= \arctan\left(\dfrac{\sqrt{R_{f,1}^2 + R_{f,2}^2}}{f}\right) \\ \theta &= \arctan\left(\dfrac{R_{f,2}}{R_{f,1}}\right) \end{cases}$$

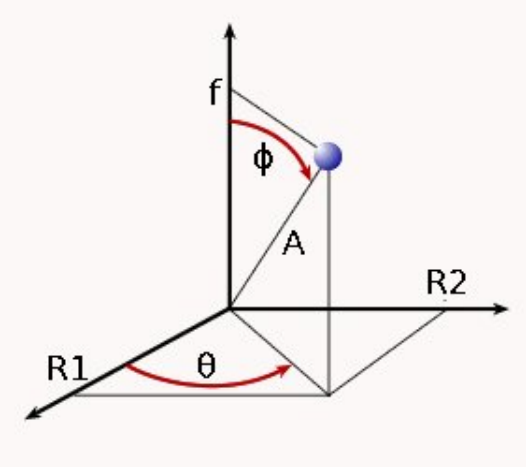


Figure 2: Représentation du signal monogène en coordonnées cartésiennes et sphériques.

1.3 Algorithme de calcul dans le domaine fréquentiel

Dans un premier temps, nous avons implémenté un algorithme naïf de calcul de la transformée de Riesz. Pour ce faire, nous nous plaçons dans le domaine fréquentiel en utilisant la FFT classique et nous appliquons l'équation (1.1) avant de retourner en représentation spatiale. Pour vérifier la validité de notre algorithme, nous présentons une première application sur une fonction sinusoïdale simple afin de pouvoir comparer nos résultats expérimentaux aux résultats théoriques.

Soit un signal $f(\vec{x}) = \cos(\vec{k}.\vec{x})$ de vecteur d'onde $\vec{k} = k_0 \begin{pmatrix} \cos(\beta) \\ \sin(\beta) \end{pmatrix}$.

D'après la définition 1.1 on peut écrire :

$$\begin{aligned} \hat{R}_f(\vec{\xi}) &= -i\frac{\vec{\xi}}{||\vec{\xi}||_2}\hat{f}(\vec{\xi}) \\ &= -\frac{i}{2}\frac{\vec{\xi}}{||\vec{\xi}||_2}(\delta(\vec{\xi}-\vec{k}) + \delta(\vec{\xi}+\vec{k})) \\ &= -\frac{i}{2}\frac{\vec{k}}{k_0}\delta(\vec{\xi}-\vec{k}) + \frac{i}{2}\frac{\vec{k}}{k_0}\delta(\vec{\xi}+\vec{k}) \\ \hat{R}_f(\vec{\xi}) &= \begin{pmatrix} \cos(\beta) \\ \sin(\beta) \end{pmatrix} \frac{i}{2}\left(\delta(\vec{\xi}+\vec{k}) - \delta(\vec{\xi}-\vec{k})\right) \end{aligned}$$

On identifie la transformée de Fourier d'un sinus. En appliquant la transformée de Fourier inverse, la transformée de Riesz de f est donc :

$$R_f(\vec{x}) = \begin{pmatrix} \cos(\beta) \\ \sin(\beta) \end{pmatrix} \sin(\vec{k}.\vec{x})$$

Comparaison des résultats théoriques et expérimentaux :

Nous avons représenté en *figure 3* les résultats obtenus sur la fonction f pour $k_0 = 5$ et $\beta = 0.4$ rad $\simeq 23°$. Les valeurs calculées pour $R_{f,1}$ et $R_{f,2}$ sont proches des valeurs théoriques sauf sur les bords de l'image. En effet, nous avons considéré l'image miroir pour les calculs aux bords alors que notre image n'est pas périodique. Néanmoins, les effets de bord seront moins présents par la suite lorsque l'on considèrera de véritables images. Nous avons donc gardé la représentation miroir plutôt adaptée dans le cas général. Numériquement, le tableau suivant donne une idée de la précision des calculs (avec les bords de l'image) :

Erreur inférieure à	Pourcentage de pixels dont l'erreur est sous le seuil
0.5	96%
0.1	76%
0.05	67%

D'autre part, on peut remarquer que $R_{f,1}$ privilégie les détails selon l'horizontale tandis que $R_{f,2}$ privilégie les détails selon la direction verticale.

Les résultats pour l'orientation et la phase locales sont également bons mis à part les effets de bord. Nous n'avons pas représenté l'amplitude qui, comme prévu, est très proche de 1 sur l'ensemble de l'image.

D'autres résultats visuels du calcul de la transformée de Riesz par FFT sont donnés en **Annexe**.

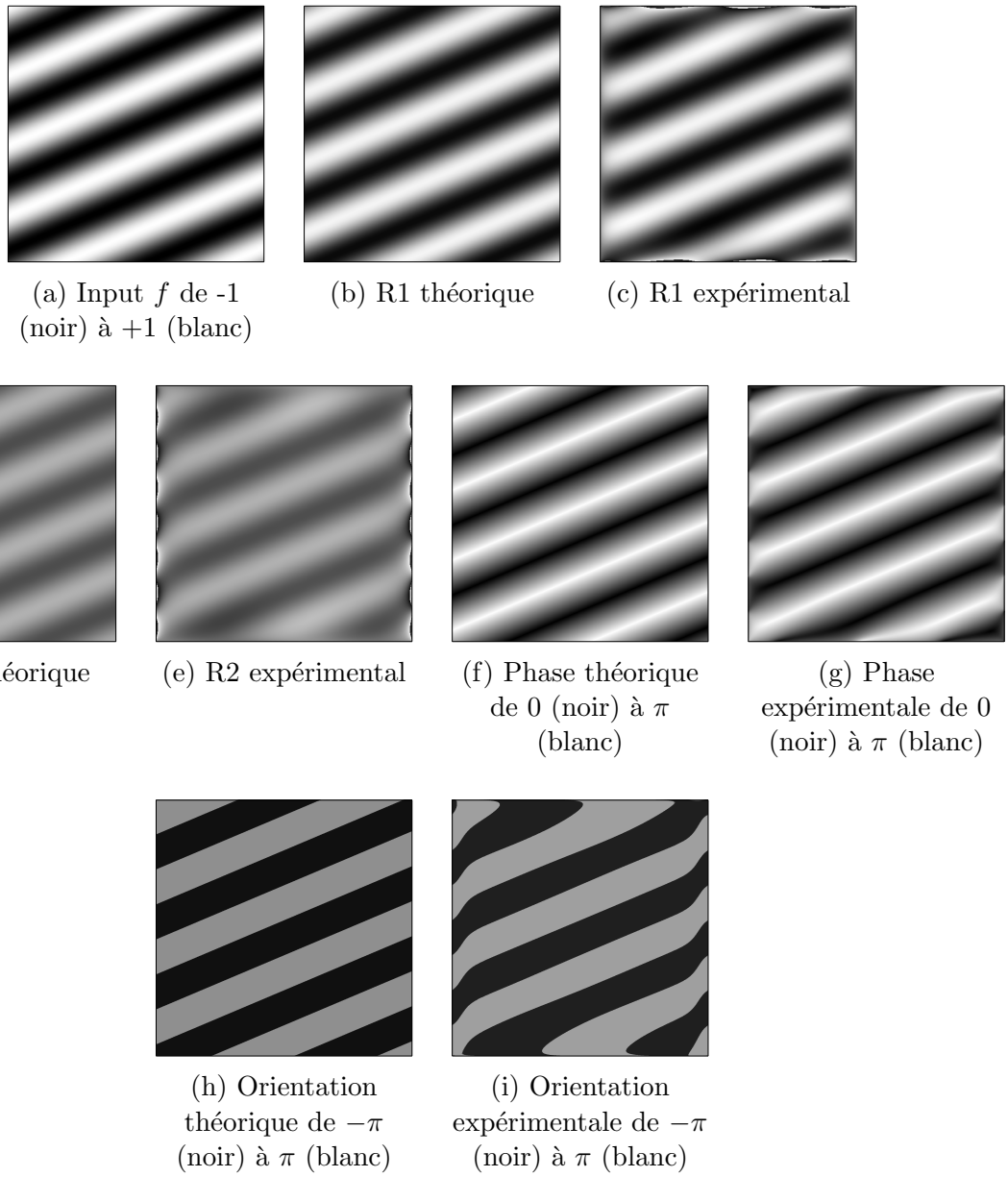
1.4 Amélioration de la carte d'orientation locale

Dû à l'équation (1.1), l'orientation locale est mal définie pour des fréquences proches de 0 et on peut tomber sur des instabilités. Pour cette raison, nous avons lissé la carte d'orientation en resituant chaque pixel dans son contexte [2].

Pour chaque pixel (m, n) de l'image, on applique :

$$\theta'_{m,n} = \frac{\sum\limits_{(i,j)\in V(m,n)} \sin^2(\Phi_{i,j})\theta_{i,j}}{\sum\limits_{(i,j)\in V(m,n)} \sin^2(\Phi_{i,j})}$$

où $V(m, n)$ est un voisinage du pixel.



(a) Input f de -1 (noir) à +1 (blanc)

(b) R1 théorique

(c) R1 expérimental

(d) R2 théorique

(e) R2 expérimental

(f) Phase théorique de 0 (noir) à π (blanc)

(g) Phase expérimentale de 0 (noir) à π (blanc)

(h) Orientation théorique de $-\pi$ (noir) à π (blanc)

(i) Orientation expérimentale de $-\pi$ (noir) à π (blanc)

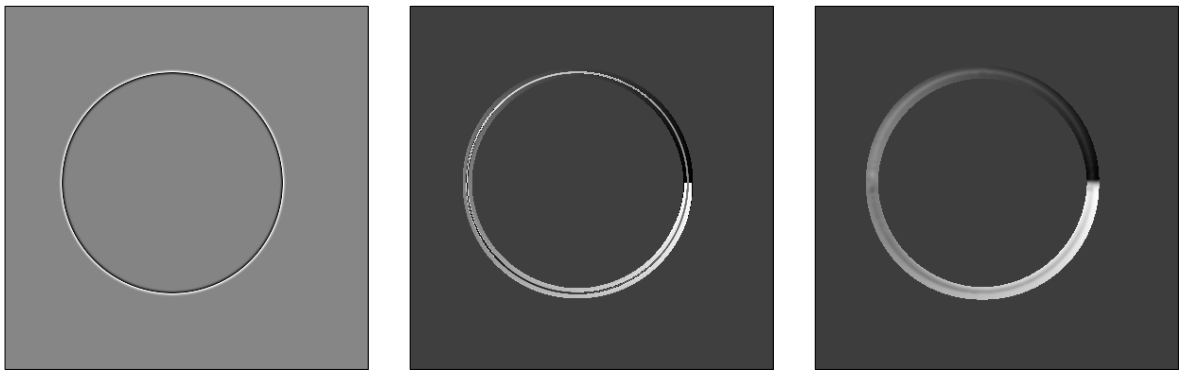
Figure 3: Transformée de Riesz sur le signal $f(\vec{x}) = \cos(\vec{k}.\vec{x})$ et comparaison avec les résultats théoriques

Ainsi, plus l'orientation du pixel voisin est fiable ($\sin^2(\Phi_{i,j})$ élevé), plus il prend d'importance dans le calcul de la nouvelle orientation $\theta'_{m,n}$. De plus, le pixel voisin n'est pris en compte que lorsque la phase locale est suffisament élevée pour que l'orientation y soit significative, c'est à dire si :

$$\sin^2(\Phi_{i,j}) > \tau \text{ où } \tau \in [0;1]$$

En pratique, nous prenons un voisinage de taille 5 et $\tau \simeq 0.2$. La *figure 4* représente l'orientation des hautes fréquences d'un disque avant et après lissage.

D'autre part, la représentation en niveau de gris de l'orientation est peu appropriée lorsque l'image considérée est relativement complexe. En effet, elle ne peut pas être



(a) Input f (b) Orientation non lissée (c) Orientation lissée

Figure 4: Orientation locale de f, sous-bande des haute-fréquences d'un disque

représentée de manière continue : la discontinuité noir/blanc entre les orientations Π et $-\Pi$ est inévitable. Pour ce faire, nous avons implémenté une mise en couleur de la carte d'orientation avec l'encodage de la *figure 5*. On peut observer en *figure 6* la différence entre ces deux représentations.

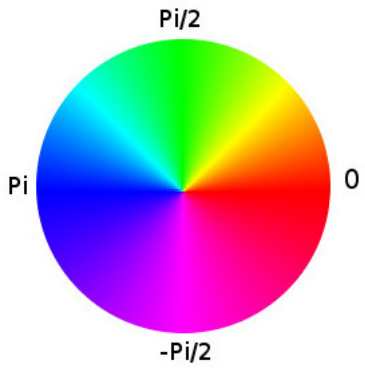


Figure 5: Encodage couleur de notre algorithme pour la représentation de l'orientation locale

(a) Input (b) Carte d'orientation en noir et blanc (c) Carte d'orientation en couleur

Figure 6: Orientation locale sur une empreinte digitale

1.5 Limites de l'algorithme par FFT

Si ces premiers résultats sont satisfaisants, effectuer le calcul de la $\mathcal{TR}$ dans le domaine fréquentiel possède deux limites essentielles :

(i) Pour le traitement des vidéos en temps réel, la FFT s'avère coûteuse (coût en $\mathcal{O}(n \log n)$). Nous souhaitons implémenter un algorithme de coût linéaire.

(ii) La transformée de Fourier nécessite de connaître l'image dans son intégralité ce qui n'est pas toujours le cas en pratique (en imagerie médicale par exemple).

Par conséquent, nous détaillons dans la partie suivante un algorithme plus rapide se basant sur une décomposition multi-échelle de l'image.

2 Algorithme rapide de calcul de la transformée de Riesz

L'objectif de cette partie est d'effectuer une approximation de la transformée de Riesz dans le domaine spatial afin d'éviter le coût lié à la transformée de Fourier.

2.1 Décomposition de l'image en sous-bandes fréquentielles

Nous commençons par décomposer l'image en une pyramide semblable à la pyramide Laplacienne proposée par P. Burt et E. Adelson [3] pour la compression des images : l'image est décomposée en un composant basse-fréquences (application d'un filtre passe-bas) et en un composant haute-fréquences (différence de l'image originale et du composant passe-bas). Un pixel étant étroitement corrélé aux pixels voisins, l'information est considérée redondante. Pour une meilleure performance de calcul par la suite (ou une meilleure compression suivant l'objectif), le composant basse-fréquences est donc décimé en supprimant un pixel sur deux sur chaque dimension. Puis le procédé est réitéré sur la nouvelle image obtenue comme illustré en *figure 7*.

Les filtres sont calculés de sorte à ce que la pyramide soit inversible : l'image doit pouvoir être reconstituée parfaitement à partir de ses sous-bandes fréquentielles. La pyramide Laplacienne, très utile pour la compression, ne permet pas une bonne reconstruction de l'image lorsque les sous-bandes ont été modifiées. Pour cette raison, nous utilisons les filtres proposés par N. Wadhwa et al. [4] pour obtenir "une pyramide de Riesz".

La *figure 8* représente les 4 premiers niveaux de la pyramide de Riesz sur "Lena".

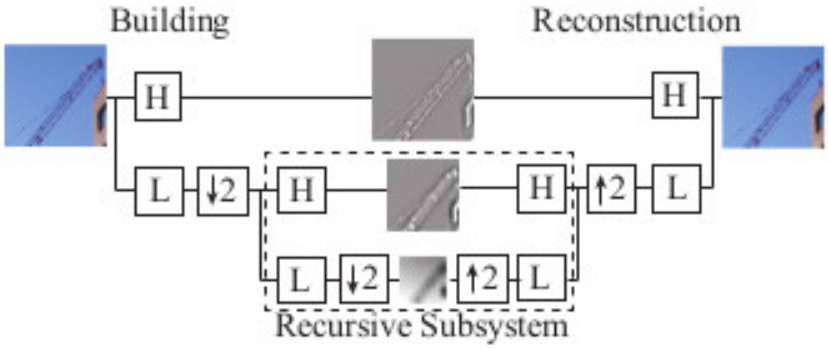
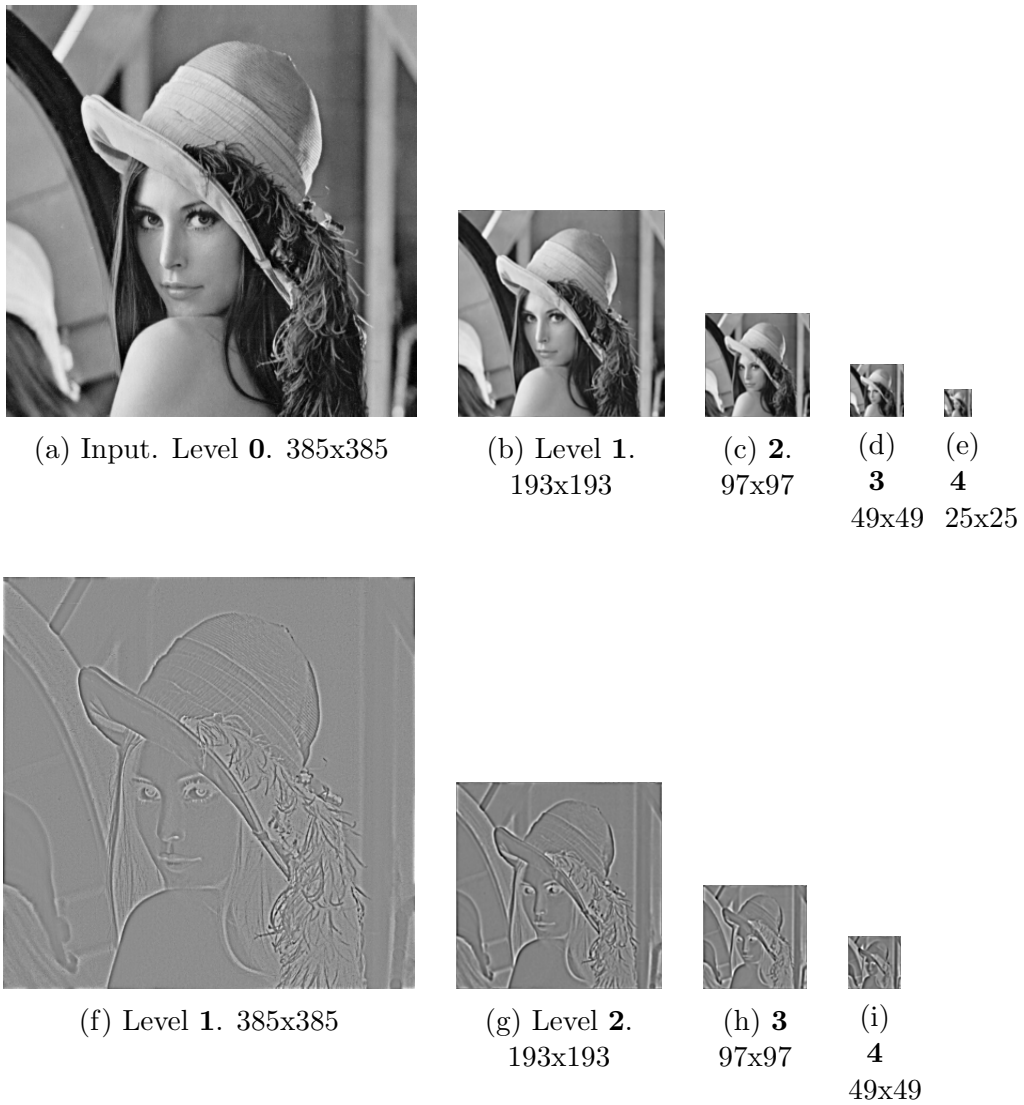


Figure 7: Décomposition multi-échelle de l'image. Les composants haute-fréquences sont identifiés par H et les basse-fréquences par L.



(a) Input. Level **0**. 385x385 (b) Level **1**. 193x193 (c) **2**. 97x97 (d) **3** 49x49 (e) **4** 25x25

(f) Level **1**. 385x385 (g) Level **2**. 193x193 (h) **3** 97x97 (i) **4** 49x49

Figure 8: Les 4 premiers niveaux de la pyramide de Riesz. La première ligne correspond aux composants basse-fréquences tandis que la seconde correspond aux composants haute-fréquences.

2.2 Approximation de la transformée de Riesz

Nous nous plaçons sur une sous-bande fréquentielle calculée précédemment. Sur cette sous-image :

$$||\vec{\xi}||_2 \simeq C,\ C \in \mathbb{R}$$

Dès lors, l'équation (1.1) s'écrit :

$$\hat{R}_f(\vec{\xi}) = C' i\vec{\xi}\hat{f}(\vec{\xi}), C' \in \mathbb{R} \tag{2.1}$$

La transformée de Riesz peut donc être approximée par le gradient à une constante multiplicative près. Dans le domaine spatial, une bonne manière de calculer le gradient est d'utiliser la formule de Taylor ce qui revient à utiliser les filtres :

$$[-0.5,0,0.5] \text{ et } [-0.5,0,0.5]^T$$

3 Comparaison des deux approches

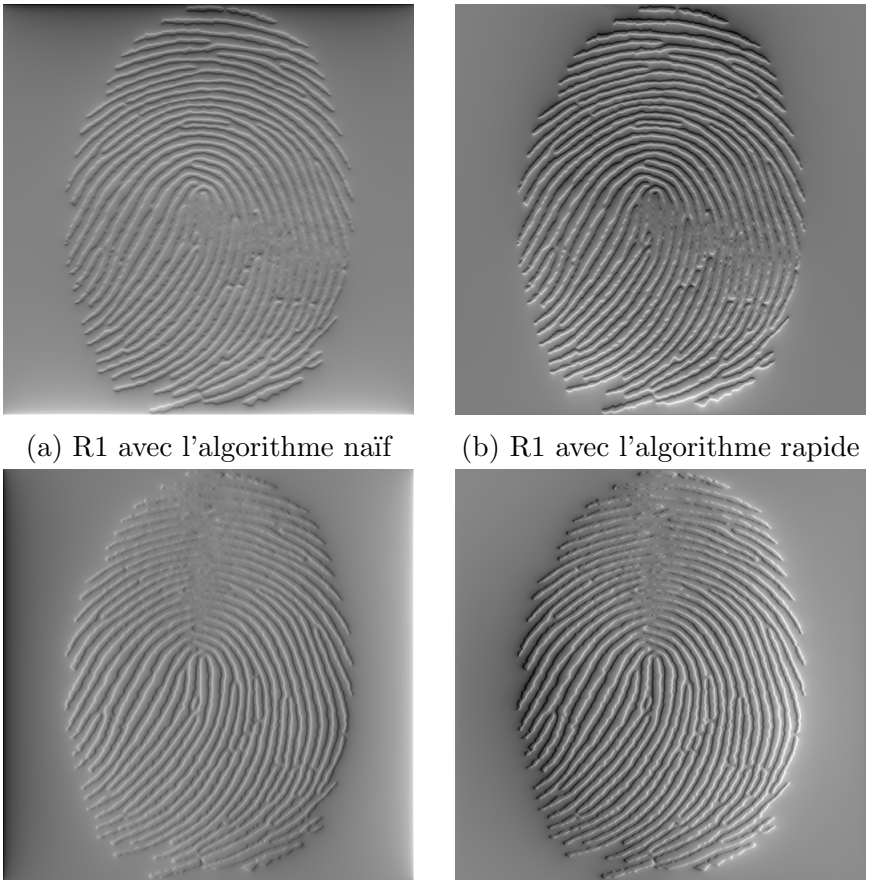
3.1 Comparaison qualitative

La *figure 9* montre les résultats obtenus avec notre nouvel algorithme sur une empreinte digitale. Ils sont sensiblement les mêmes que ce que l'on avait obtenu avec la FFT au signe près. En effet, la constante multiplicative de l'équation (2.1) est négative d'après la définition de la transformée de riesz (1.1).

Nous n'avons pas réussi à faire de comparaison numérique pertinente des valeurs des composantes de la transformée de Riesz entre les deux algorithmes.

3.2 Comparaison en performance

Nous avons représenté en *figure 10* le temps de calcul de la $\mathcal{TR}$ des deux algorithmes en fonction du nombre d'images traitées. On obtient bien un profil linéaire pour l'agorithme par décomposition multi-échelle. Le gain en performance est non négligeable surtout si l'on traite des vidéos. En *figure 11*, nous avons représenté le temps de calcul de notre algorithme d'amplification de mouvement sur une vidéo avec les deux méthodes. Pour une vidéo d'environ 10 secondes (soit 200 frames traités), l'algorithme rapide prend 2 minutes tandis que le passage par Fourier prend 10 minutes. En parallélisant les calculs et en utilisant une machine puissante, il est possible de se ramener à du temps réel avec l'algorithme rapide.



(a) R1 avec l'algorithme naïf (b) R1 avec l'algorithme rapide

(c) R2 avec l'algorithme naïf (d) R2 avec l'algorithme rapide

Figure 9: Comparaison de la transformée de Riesz obtenue avec l'algorithme naïf d'une part, et l'algorithme rapide d'autre part

4 Amplification de mouvement dans les vidéos

4.1 Gestion des vidéos couleur en OpenCV

Afin de pouvoir tester nos applications sur des vidéos, nous avons d'abord implémenté un gestionnaire de vidéo qui permet de récupérer les images successives d'une vidéo, de les traiter, et de créer une nouvelle vidéo dans laquelle le mouvement a été amplifié.

Pour pouvoir traiter directement des vidéos couleur, nous utilisons la représentation Y-Cb-Cr. L'information envoyée à notre algorithme est la luminance Y, grandeur correspondant à la sensation visuelle de luminosité. Les deux composantes de chrominance Cb et Cr permettent de reconstituer la couleur du pixel, mais ne sont pas utiles dans le processus d'amplification de mouvement.

4.2 Amplification de mouvement sur une fonction sinusoïdale

Nous présentons ici une méthode d'amplification de mouvement basée sur la phase qui donne de bons résultats sur de petits mouvements. L'objectif est d'être capable d'amplifier

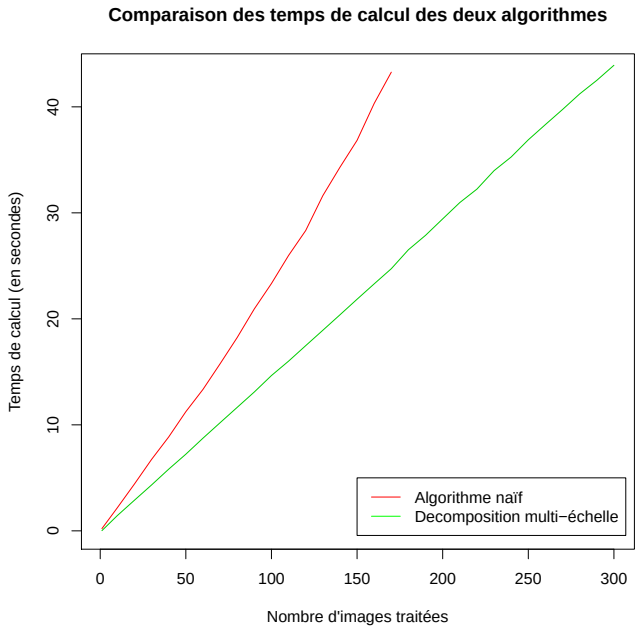


Figure 10: Comparaison des performances des deux méthodes sur le calcul de la transformée de Riesz

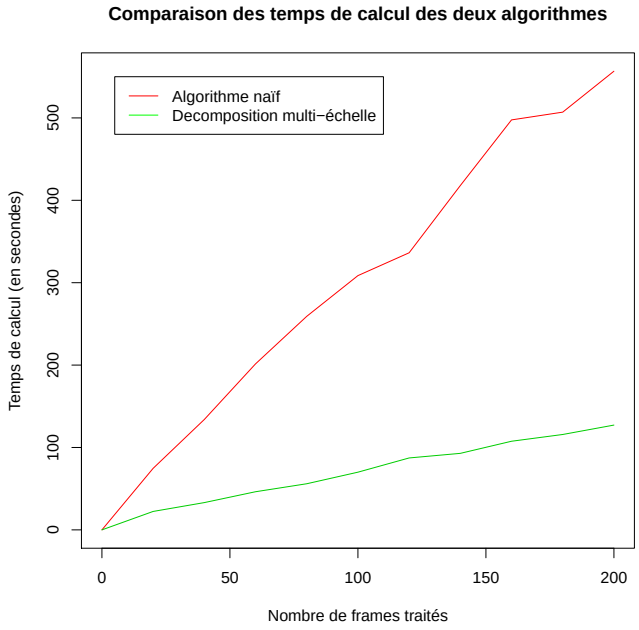


Figure 11: Comparaison des performances des deux méthodes dans l'algorithme d'amplification de mouvement dans les vidéos

des détails imperceptibles au premier abord tels que la respiration d'un être humain. Pour en comprendre le principe, commençons par traiter un exemple simple

Soit $f(x, y, t) = Acos(w_x(x + \delta(t)) + w_y y)$ une fonction sinusoïdale avec un léger

mouvement $\delta(t)$ selon l'axe des x (vertical dans nos images).

La transformée de Riesz de f (déjà calculée en section 2.3) est :

$$R_f(\vec{x}) = A\frac{(w_x, w_y)}{\sqrt{w_x^2 + w_y^2}} sin(w_x(x + \delta(t)) + w_y y)$$

En calculant la phase locale de f on obtient donc :

$$\Phi = w_x(x + \delta(t)) + w_y y$$

On effectue une différence de phase entre deux images successives de la vidéo afin d'isoler le terme de la phase qui dépend du temps.

$$\text{Différence de phase : } \Phi_0 = w_x \delta(t)$$

On amplifie ensuite Φ_0 par un coefficient α.

Or la phase locale Φ peut être vue comme la phase du nombre complexe :

$$Ae^{i\Phi} = I + iQ$$

La fonction f peut alors être décalée dans le sens de l'orientation dominante en réinjectant la phase amplifiée $\alpha\Phi_0$ de la manière suivante :

$$\text{Réel}(e^{-i\alpha\Phi_0}(f + iQ))$$

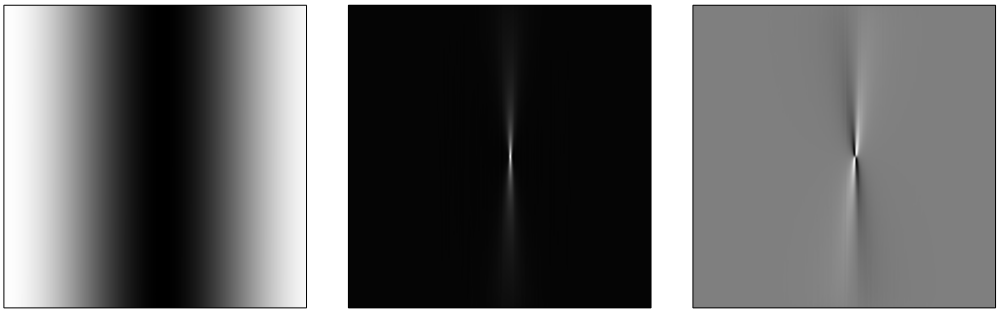
Nous obtenons une sinusoïde dont le mouvement est amplifié :

$$Acos(w_x(x + (1 + \alpha)\delta(t)) + w_y y)$$

Nous avons représenté en *figure 12* les résultats obtenus sur un cosinus vertical. On remarque que le résultat de l'algorithme rapide (f) est meilleur que celui de l'algorithme naïf (e) : l'amplification du mouvement est plus visible et l'image de sortie est moins atteinte par le bruit alors que Φ_0 a été lissée par une gaussienne dans (e).

Remarque : Dans l'algorithme rapide, l'amplification de la phase est effectuée à chaque niveau de la pyramide et l'image est reconstituée après modification de tous les étages de la pyramide.

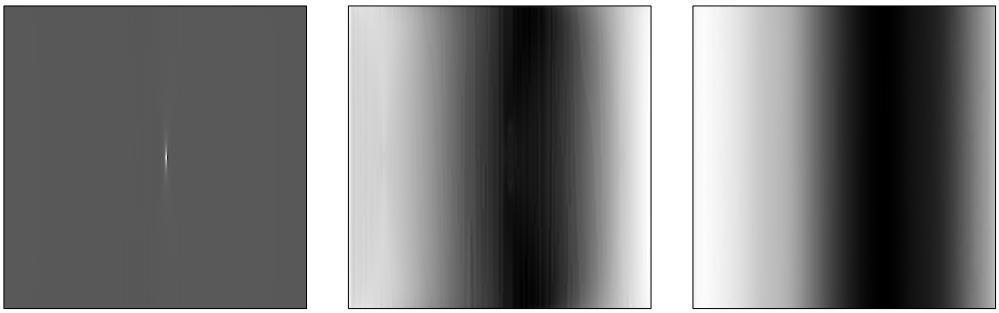
Résultats en vidéo : Des vidéos sont fournies dans le dossier *./videos/*. Pour chacune d'entre elle vous pouvez observer la vidéo originale qui présente un léger mouvement dans une direction donnée.



(a) Cosinus vertical

(b) Différence de $\Phi cos(\theta)$ entre les deux images

(c) Différence de $\Phi sin(\theta)$ entre les deux images



(d) Différence de phase Φ_0 résultante, i.e. la phase du mouvement

(e) Amplification par l'algorithme FFT avec α=50 et avec un lissage gaussien

(f) Amplification par l'algorithme rapide avec α=50 sans lissage supplémentaire

Figure 12: Le cosinus (a) est décalé à droite d'un pixel pour créer un léger mouvement horizontal. Le calcul de la phase du mouvement en (b) et (c) donne la figure (d). La phase est ensuite amplifiée d'un facteur α=50. On obtient la figure (e) avec l'algorithme naïf utilisant la FFT et la figure (f) en utilisant l'algorithme rapide.

5 Références

[1] http://people.csail.mit.edu/nwadhwa/riesz-pyramid/

[2] T. Bülow, D. Pallek and G. Sommer. Riesz transforms for the isotropic estimation of the local phase of Moiré interferograms, 2000.

[3] P. Burt and E. Adelson. The laplacian pyramid as a compact image code. IEEE Trans. Commun., 1983.

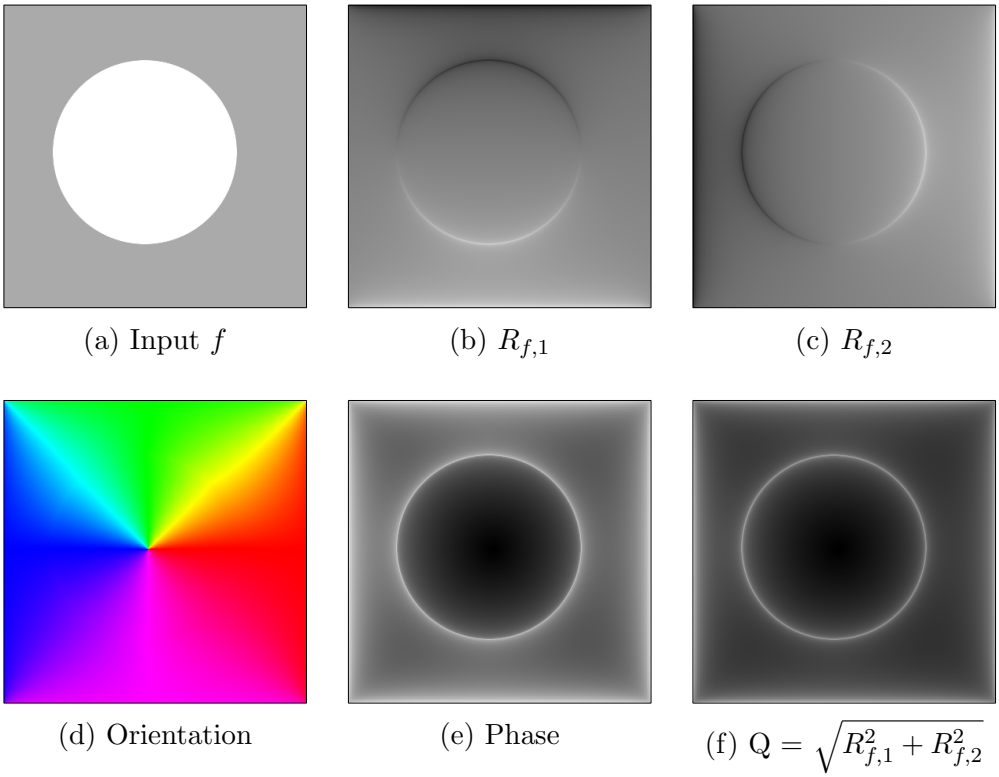
[4] N. Wadhwa, M. Rubinstein, F. Durand, and W. Freeman. Supplemental for Riesz pyramids for fast phase-based video magnification.

[5] N. Wadhwa, M. Rubinstein, F. Durand, and W. Freeman. Phase-based video motion processing. ACM Trans. Graph., 2013.

[6] N. Wadhwa, M. Rubinstein, F. Durand, and W. Freeman. Riesz Pyramids for Fast Phase-Based Video Magnification, ICCP 2014.

[7] N. Wadhwa, M. Rubinstein, F. Durand, and W. Freeman. Quaternionic representation of the riesz pyramid for video magnification. Technical report, MIT Computer Science and Artificial Intelligence Laboratory, 2014.

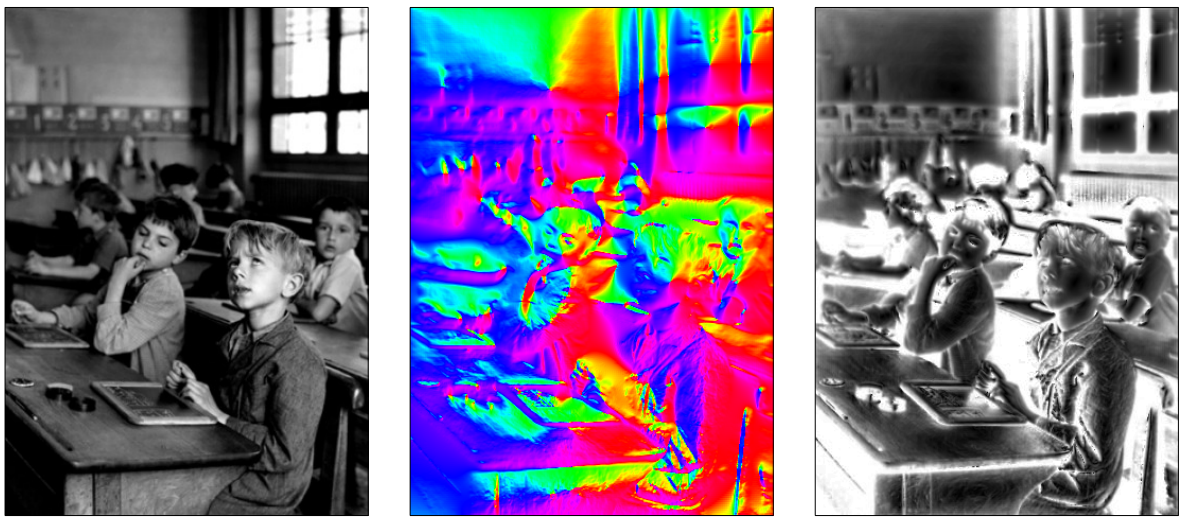
6 Annexe : analyse des caractéristiques locales des images par transformée de Riesz



(a) Input f (b) $R_{f,1}$ (c) $R_{f,2}$

(d) Orientation (e) Phase (f) Q $= \sqrt{R_{f,1}^2 + R_{f,2}^2}$

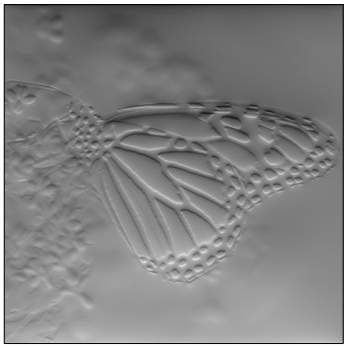
Figure 13: Caractéristiques d'une forme géométrique simple illustrant la signification des composantes du signal monogène



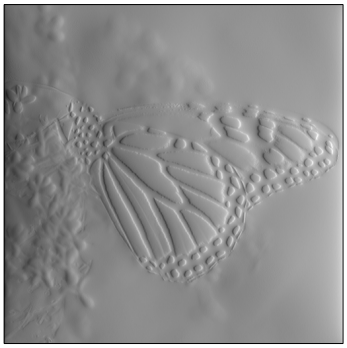
(a) Input (b) Orientation (c) Phase

Figure 14: Caractéristiques locales sur une photographie de Robert Doisneau

(a) Input

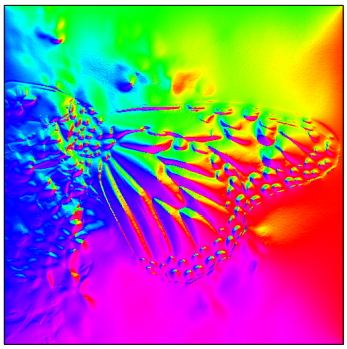


(b) $R_{f,1}$

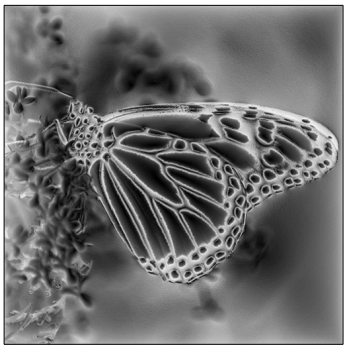


(c) $R_{f,2}$

(d) Orientation de $-\pi$ (noir) à π (blanc)

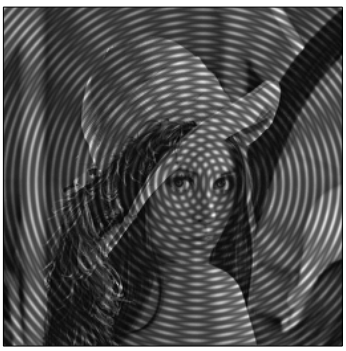


(e) Orientation couleur

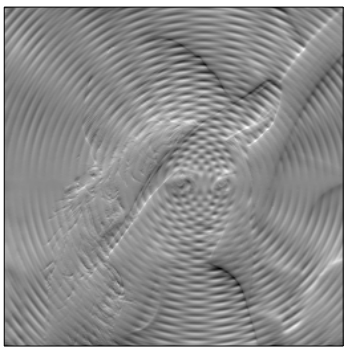


(f) Phase

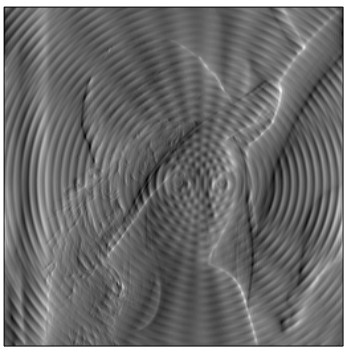
Figure 15: Etude sur un papillon



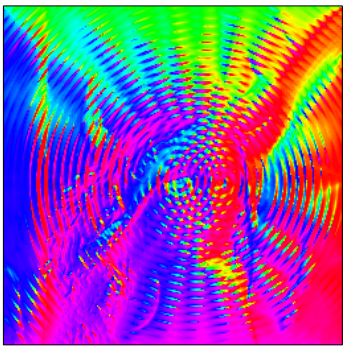
(a) Input



(b) $R_{f,1}$

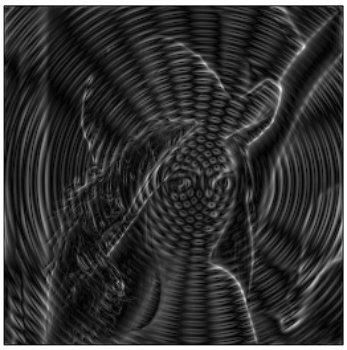


(c) $R_{f,2}$



(d) Orientation

(e) Phase



(f) Q = $\sqrt{R_{f,1}^2 + R_{f,2}^2}$

Figure 16: Etude sur 'Lena psychédélique'