Imagerie sous-pixellique

Rapport de projet

Jean Feydy
École Normale Supérieure

Vincent Vidal
École Normale Supérieure

Résumé : Il s'agit de proposer une méthode de visualisation (interactive ou non, au choix) des lignes de gradient d'une image. On essaiera autant que possible de commencer par analyser les difficultés (par exemple le choix des lignes visualisées), et de discuter quelques solutions possibles pour les contourner. On considérera plusieurs possibilités d'interpolation, et on pourra s'inspirer du module `llview`.

Table des matières

Introduction

Disposant d'une image bitmap $I : [\![1,\, N]\!] \times [\![1,\, M]\!] \to \mathbb{R}$, nous souhaitons calculer ses lignes de gradient – les courbes paramétrées L qui vérifient, avec f une interpolation continue de I :

$$\forall t \in \mathbb{R}, \quad \frac{\mathrm{d}L(t)}{\mathrm{d}t} = \nabla f(L(t))\,. \tag{1}$$

Ce travail peut être décomposé comme suit : Dans un premier temps, section 1, il s'agit d'estimer le gradient ∇f de I en tout point du domaine continu. À cette fin, nous commencons par appliquer à I un prétraitement qui limitera l'influence du bruit sur le résultat final. On calcule ensuite les valeurs du gradient sur le domaine *discret* de définition de I, par convolution circulaire avec la dérivé d'un filtre gaussien, avant d'appliquer un lissage, possiblement anisotrope. Pour finir, on interpole ces valeurs du gradient sur le domaine *continu*.

Dans un second temps, section 2, on cherche à résoudre numériquement l'équation 1 : on utilisera les habituels schémas de Cauchy et Runge-Kutta, ainsi qu'une méthode d'intégration explicite originale, dans le cas ou le gradient est supposé constant par morceaux.

Enfin, section 3, on s'intéresse à la manière d'afficher ces lignes de gradient à l'écran : le choix des lignes et des paramètres graphiques y est étudié.

1 Estimation du gradient

Prétraitement de l'image

Avant d'appliquer un opérateur de calcul du gradient, il convient de lisser notre image : une ligne de gradient étant piégée dans la première cuvette de potentiel rencontrée, un faible niveau de bruit suffit à grandement limiter la longueur des lignes intégrées et, par là, la lisibilité de la visualisation finale. Nous prendrons donc la précaution d'appliquer un filtre gaussien ou médian à notre image.

Calcul du gradient

Pour calculer numériquement le gradient, nous nous contentons de convoler l'image lissée avec un filtre de Sobel (D_x, D_y). Par associativité du produit de convolution, l'opération résultante du lissage gaussien et du filtre de Sobel est équivalente à la convolution par la dérivée discrète d'une gaussienne.

$$D_x = \frac{1}{8}\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \qquad D_y = \frac{1}{8}\begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \tag{2}$$

Lissage du gradient

On peut appliquer au gradient les mêmes opérations de lissage qu'à l'image initiale : filtrages médian et par convolution, ce qui réduira encore le nombre de points d'annulation du gradient – i.e. d'arrêts forcés de la ligne. D'autres méthodes – fortement anisotropes et spécifiques aux champs de vecteurs – sont développées dans [2] et [5] mais nous ne les avons pas implémenté ici.

Interpolation

Pour estimer notre champ sur le domaine continu $[0,\, N] \times [0,\, M]$, nous nous sommes contenté ici des méthodes d'interpolation constante par morceaux et bilinéaires, qui donnent des résultats corrects tout en mettant bien en évidence l'influence du choix de la méthode sur la précision du résultat final.

Bilan

Pour des raisons de compréhension, nous avons donc fractionné le calcul du champ de gradient en cinq étapes : lissage médian de l'image, lissage par convolution, calcul du gradient par différences finies, lissage du gradient par convolution et, éventuellement, lissage médian du gradient (sur les deux composantes, horizontale et verticale). Si I est l'image initiale, on calcule donc le gradient (G_x, G_y) comme suit :

$$I = \mathrm{Med}(I) \tag{3}$$
$$G_x = C_g * D_x * C_i * I \tag{4}$$
$$G_x = \mathrm{Med}(G_x), \tag{5}$$

avec C_g, C_i filtres de débruitage standards, gaussiens ou uniformes.

2 Intégration du gradient

Schéma d'intégration des lignes

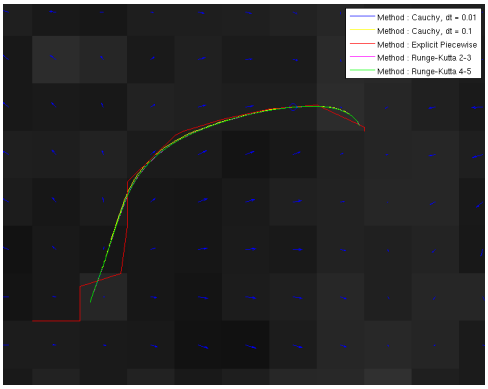
Nous disposons maintenant d'un champ de gradient sur l'image : encore faut-il l'intégrer. À cet effet, nous utilisons simplement les méthodes de Cauchy et de Runge-Kutta, directement implémentées dans Matlab. Les résultats obtenus sont alors intéressants, mais le temps nécessaire au calcul de lignes de taille "macroscopique" est quelque peu décourageant : les méthodes de Runge-Kutta, aux courbes lisses, utilisent de nombreux points par pixel, tandis que les méthodes de Cauchy ont parfois tendance à zigzaguer le long des arêtes sans vraiment avancer. Notre objectif étant à terme de pouvoir tracer des centaines de lignes longues (∼100px) par image, nous avons donc implémenté, à la main, une résolution explicite du problème lorsque le champ est constant par morceau sur les pixels : dans la suite, on la nommera *Explicit-Piecewise*.

Comparaison des méthodes

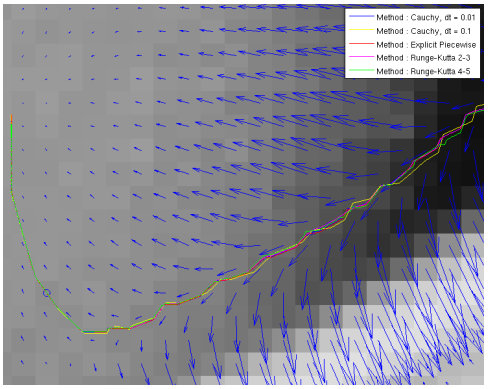
Étude qualitative Il n'y a, à première vue, aucune surprise dans l'apparence des lignes intégrées, Figure 1 : la méthode Explicit-Piecewise fait bien ce que l'on attend d'elle, les schémas de Cauchy semblent bien converger vers la solution donnée par la méthode de Runge-Kutta... Les plus graves erreurs apparaissent lorsque le champ de gradient change radicalement de direction entre deux pixels voisins : une faible incertitude peut alors mener à une bifurcation des trajectoires, comme sur la Figure 1d.

Tous ces renseignements, s'ils sont pertinents, n'en restent pas moins fort limités : nous manquons ici de lignes de référence, auxquelles comparer nos solutions numériques.

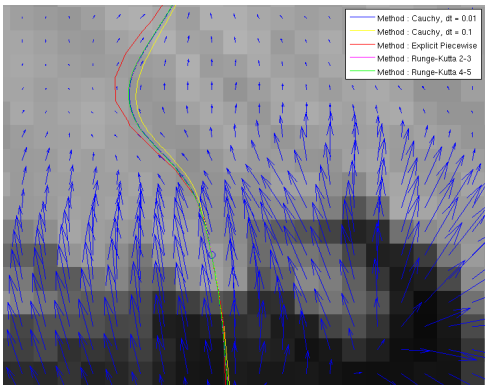
Étude quantitative Pour continuer notre étude, nous nous concentrons donc sur des images bien particulières, idéales, dont les lignes de gradient sont connues : une image *radiale* et une image *courbée*, Figure 2. Les résultats attendus sont, respectivement, des rayons et des cercles concentriques, mais des erreurs peuvent venir s'imiscer dans les deux étapes du calcul. D'une part, l'image



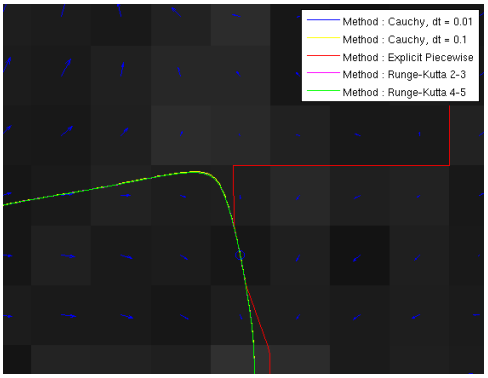
(a) Les trois méthodes d'intégration des lignes.



(b) Si le courant est fort et régulier, les écarts sont contrôlés.



(c) Des erreurs peuvent survenir lorsque le pas de discrétisation est trop grand, qu'il soit spatial (courbe rouge) ou temporel (courbe jaune).



(d) Un courant trop faible peut conduire à de regrettables bifurcations.

FIGURE 1 – Dans ces quatres images, le gradient est interpolé de façon bilinéaire entre les points de la grille.

étant discrétisée sur une grille, le champ de gradient calculée ne correspondra pas nécessairement au champ théorique, tant sur les points de la grille que sur les valeurs interpolées. D'autre part, nos méthodes d'intégration n'étant qu'approchées, elle peuvent aussi induire des erreurs.

Afin de ne pas prendre en compte les éventuels décalages de phase, ou reparamétrisations de lignes, on définit l'erreur moyenne de la ligne de champ L de longueur l à la courbe théorique L_T comme

$$\mathrm{Err}(L, L_T) = \frac{1}{l}\int_0^l \mathrm{dist}_2(L(s), L_T)\mathrm{d}s, \qquad (6)$$

avec L paramétrée par longueur d'arc, et $\mathrm{dist}_2(x, L_T)$ la distance euclidienne entre le point x et la courbe paramétrée L_T.

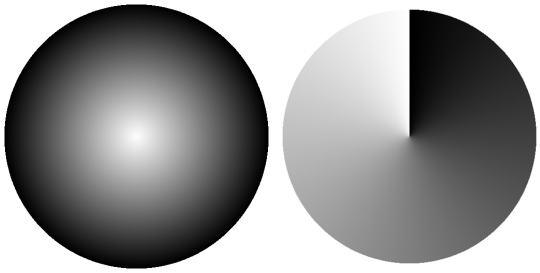
Étude d'une image radiale

Formulation mathématique Dans un repère centré, en coordonnées polaires, on définit l'intensité de notre image I par :

$$I(\rho, \theta) = \rho\frac{255}{R}, \qquad (7)$$

$$\nabla I(\rho, \theta) = \frac{255}{R}\vec{u}_\rho, \qquad (8)$$

avec $R = 255$, par exemple. Cette image nous permet de connaître les directions privilégiées de notre



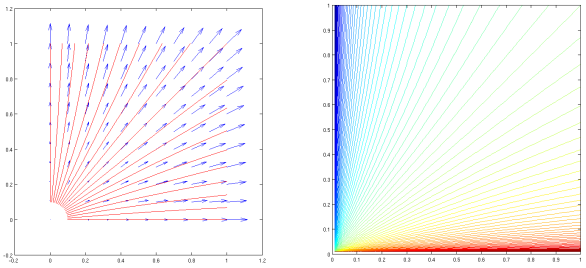
(a) Champ radial. (b) Champ courbe.

FIGURE 2 – Figures test représentant les deux types locaux de gradient.

algorithme et résume, en un test, l'information que nous aurions obtenu en appliquant ce calcul à des gradients "linéaires" de directions variables.

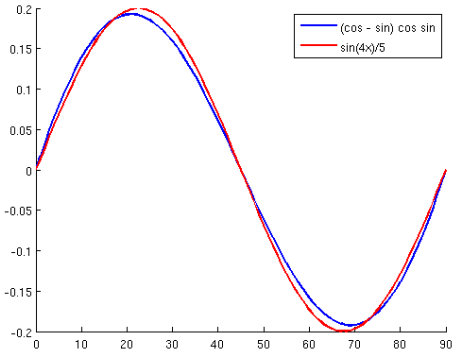
Résultats expérimentaux Les courbes d'erreurs obtenues en tracant des lignes partant d'un cercle de rayon fixé, Figure 14, sont très intéressantes. Tout d'abord, on remarque que l'erreur dépend uniquement de la valeur de l'angle modulo 90° : on retrouve donc bien l'invariance par rotation droite du problème. Les symétries par rapport aux multiples de 45° reflètent, elles, les invariances par des symétrie axiales de directions les axes du repères, la première et la deuxième bissectrices. On ne trace donc les courbes d'erreurs que pour des angles θ variant entre 0° et 90°, et l'annulation de l'erreur aux angles multiples de 45° correspond aux axes de symétrie du repère.

En dehors de ces simples vérification, deux leçons sont à retenir de nos résultats. La première, c'est la relative imprécision du filtrage médian, qu'il soit appliqué au gradient ou à l'image : si les résultats sont parfois satisfaisants à une échelle macroscopique, il vaut mieux s'en tenir éloigné si l'on recherche une précision sous-pixellique. La deuxième, c'est l'intérêt d'une interpolation fine du gradient, qui évite les bifurcations brutales. Alors qu'un gradient constant par morceaux induit de fortes déviations pour des directions presque – mais pas exactement – verticales, le phénomène est complètement maîtrisé avec une interpolation bilinéaire du gradient : l'erreur semble évoluer en $|\sin(4\theta)|$. C'est alors que nos erreurs de tracé sont les plus faibles, de l'ordre du millième de pixel.



(a) Le flot bilinéaire, et quelques lignes de champ intégrées par Matlab .

(b) Lignes de niveau de la fonction F.



(c) Notre erreur ressemble fort à $\sin(4\theta)$.

FIGURE 3 – Étude des lignes de champ pour un flot bilinéaire.

Interprétation du $|\sin(4\theta)|$ Cette apparente simplicité de l'erreur contraste avec les autres résultats, plus difficiles à analyser. Tâchons donc de comprendre l'origine de cette courbe, en utilisant un modèlé simplifié. Plutôt que de considérer une discrétisation fine du champ de vecteur, supposons simplement disposer sur $[0,1] \times [0,1]$ d'un champ $X(x,y) = \begin{pmatrix} u(x,y) \\ v(x,y) \end{pmatrix}$ obtenu par interpolation bilinéaire entre les quatres coins du carré :

$$X(0,1) = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad X(1,1) = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \tag{9}$$

$$X(0,0) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad X(1,0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}. \tag{10}$$

On a donc, avec $a = 1/\sqrt{2}$:

$$X(x,y) = \begin{pmatrix} x(1+(a-1)y) \\ y(1+(a-1)x) \end{pmatrix}. \quad (11)$$

Intégrer nos lignes de champ revient donc à résoudre une équation de Lotka-Volterra, comme utilisées en écologie pour décrire les systèmes proies-prédateurs. On remarque alors qu'avec $F(x,y) = (1-a)(y-x) + \log(x/y)$, X est en tout point orthogonal au gradient de F : c'est donc que nos solutions évoluent à F constant, et que nos lignes de gradient sont en fait les lignes de niveau de F, tracée figure 3b.

Comment, alors, évaluer l'erreur induite par la non radialité du champ ? En coordonnées cylindriques, partant d'un point $p_0 = (\rho = r, \theta = \theta)$, et suivant une ligne de champ "théorique", radiale, on devrait croiser le cercle de rayon R au point $p_T = (\rho = R, \theta = \theta)$. Malheureusement, le champ imparfait induit une dérive, et le cercle est en réalité croisé en un point $p_P = (\rho = R, \theta = \theta_P)$ tel que $F(p_P) = F(p_0)$. L'erreur, après une longueur de courbe l légèrement supérieure à $R - r$ est donc $\|p_T - p_P\|$. Dans l'hypothèse où cette déviation est suffisamment faible pour pouvoir linéariser F au voisinage de p_T – ce qui semble tout à fait plausible, au vu des ordres de grandeur mis en jeu – on a :

$$\nabla F(p_T) \cdot (p_P - p_T) = F(p_P) - F(p_T) \quad (12)$$
$$= F(p_0) - F(p_T). \quad (13)$$

Or, en coordonnées polaires, on a :

$$F(\rho,\theta) = (1-a)r(\sin\theta - \cos\theta) - \log(\tan\theta) \quad (14)$$

$$F(p_0) - F(p_T) = F(r,\theta) - F(R,\theta) \quad (15)$$
$$= (1-a)(R-r)(\cos\theta - \sin\theta) \quad (16)$$

$$\nabla F(\rho,\theta) = \begin{pmatrix} \frac{\partial F}{\partial r} \\ \frac{1}{r}\frac{\partial F}{\partial \theta} \end{pmatrix} \quad (17)$$
$$= \begin{pmatrix} (1-a)(\sin\theta - \cos\theta) \\ (1-a)(\cos\theta + \sin\theta) - \frac{1}{r}\frac{\tan'\theta}{\tan\theta} \end{pmatrix} \quad (18)$$
$$= \begin{pmatrix} (1-a)(\sin\theta - \cos\theta) \\ (1-a)(\cos\theta + \sin\theta) - \frac{1}{r}\frac{1}{\cos\theta\sin\theta} \end{pmatrix}. \quad (19)$$

Aussi, $p_P - p_T$ étant essentiellement tangent au cercle de rayon R en p_T, on trouve une expression approchée de $\nabla F(p_T) \cdot (p_P - p_T)$ sous la forme :

$$\left((1-a)(\cos\theta + \sin\theta) - \tfrac{1}{r}\tfrac{1}{\cos\theta\sin\theta}\right)(p_P - p_T) \cdot \vec{u}_\theta, \quad (20)$$

puis :

$$p_P - p_T \simeq (p_P - p_T) \cdot \vec{u}_\theta \quad (21)$$
$$= \frac{(1-a)(R-r)(\cos\theta - \sin\theta)}{(1-a)(\cos\theta + \sin\theta) - \frac{1}{r}\frac{1}{\cos\theta\sin\theta}}. \quad (22)$$

La dépendance en R n'affecte donc pas le profil de la courbe, qui varie d'ailleurs très peu en fonction de r, pour des faibles valeurs – r est de toutes façons compris entre 0 et 1. En effet, comme $(1-a) < 0.35$, et $\cos\theta\sin\theta < 1$, on peut en bonne approximation négliger le terme $(1-a)(\cos\theta + \sin\theta)$ devant le $\frac{1}{r}\frac{1}{\cos\theta\sin\theta}$. On a donc une erreur (signée) approximativement égale à :

$$p_P - p_T \simeq -(1-a)r(R-r)(\cos\theta - \sin\theta)\cos\theta\sin\theta, \quad (23)$$

pour θ entre 0° et 90°. Sur ce domaine, la ressemblance avec $\sin 4\theta$ est frappante (voir Figure 3c), et notre première impression s'en trouve donc bien expliquée.

Étude d'une image courbée

Erreur à courbure constante Les directions privilégiées de nos algorithme d'intégration des lignes étant maintenant clairement établies, nous sommes en mesure d'estimer la dérive induite par un champ de vecteurs constant, aux lignes de gradient droites. Nous nous intéressons donc maintenant à l'influence de la courbure du champ : un champ de gradient de faible courbure (ligne théoriques presque droites) devrait, intuitivement, se révéler plus facile à intégrer qu'un champ très courbé. Une "sortie de route" est après tout bien plus difficile à éviter dans un virage serré que sur une autoroute quasi-rectiligne.

Formulation mathématique Dans un repère centré, en coordonnées polaires, on aimerait définir l'intensité de notre image I par la simple formule :

$$I(\rho,\theta) = \theta\frac{255}{2\pi}, \quad (24)$$
$$\nabla I(\rho,\theta) = \frac{1}{\rho}\frac{255}{2\pi}\vec{u}_\theta. \quad (25)$$

Malheureusement, la décroissance du gradient en $\frac{1}{\rho}$ est ici fort malvenue, car elle induit une inéquité de traitement entre les faibles rayons de courbure d'une part, associés à des faibles valeurs de ρ, des gradients de norme raisonnable, et les forts rayons de courbure d'autre part, associés à des gradients dont la norme peut être plus petite que 1. L'influence de la seule courbure étant ici notre objet d'étude, nous avons recours à un petit artifice...

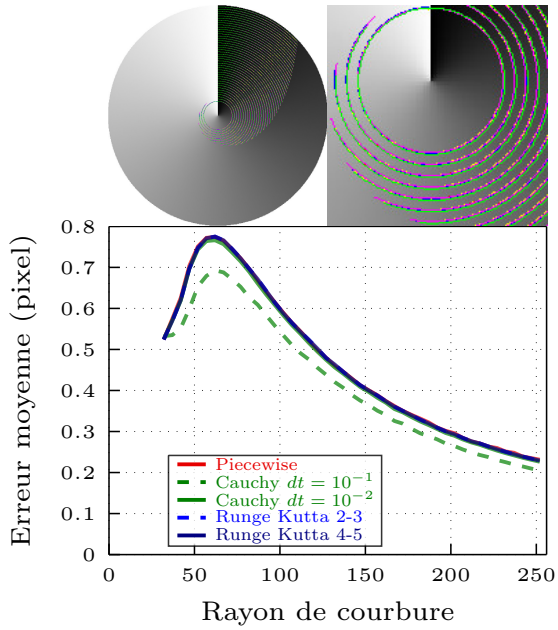


FIGURE 4 – Écarts moyens aux lignes de flot théoriques (cercles concentriques), sans prétraitement, pour un champ de vecteur purement tangentiel $\vec{u}_\theta$, en fonction du rayon de courbure R_C. L'image de fond, courbée, est donnée dans le seul but d'aider à la lecture de la figure, mais n'est en rien utilisée dans le calcul des lignes. Ces dernières sont intégrées en partant de l'axe vertical, sur une longueur de 200px.

Gradient de norme constante, explicite Pour commencer, nous court-circuitons notre algorithme, en écrivant directement le champ de "gradient" à intégrer sur la grille,

$$\nabla\tilde{I}(\rho,\theta) \quad = \quad \vec{u}_\theta. \tag{26}$$

N'étant pas de rotationnel nul, ce champ ne saurait dériver d'un potentiel. Il nous permet néanmoins de comprendre l'influence de la phase d'intégration numérique, seule, sur la dérive induite par la courbure du champ de vecteur. Toute erreur sera due à la discrétisation de ce dernier, et non à une perte d'information lors de son calcul sur les points de la grille.

La courbe obtenue, Figure 4, présente une belle décroissance de l'erreur moyenne en $1/R_C$ – la correspondance avec $R_C \mapsto 60/R_C$ est parfaite –, si ce n'est pour les faibles valeurs du rayon de courbure : nous ne nous expliquons pas ces petites erreur pour les valeurs de R_C inférieures à 50px.

En revanche, nous souhaiterions proposer une explication de la décroissance asymptotique de l'erreur en $1/R_C$. Supposons donné un rayon de courbure R, et une longueur de courbe l. On part, pour l'intégration du champ $\nabla\tilde{I}(\rho,\theta) = \vec{u}_\theta$, du point $p_0 = (\rho = R, \theta = 0) = (x = R, y = 0)$; on a alors $\nabla\tilde{I}(p_0) = 0\vec{u}_x + \vec{u}_y$. Aussi, en supposant le champ constant par morceaux sur un pixel de taille 1×1 autour de p_0, on a que la ligne intégrée suit une ligne droite, de p_0 à $p_1 = (x = R, y = 1)$. En notation complexes, on a $p_1 = \left(1 + \frac{\mathrm{i}}{R}\right) p_0$. Le point clé est alors qu'en bonne approximation, la courbe suivra de nouveau une trajectoire rectiligne, du point p_1 au point $p_2 = \left(1 + \frac{\mathrm{i}}{R}\right) p_1$, sur une longueur

$$\begin{aligned} |p_2 - p_1| &= |p_1|\left|1 + \frac{\mathrm{i}}{R} - 1\right| && (27)\\ &= |R + o(R)|\left|\frac{\mathrm{i}}{R}\right| && (28)\\ &= 1 + o(1). && (29) \end{aligned}$$

Bien entendu, cela n'est pas tout à fait exact, car $\nabla\tilde{I}(p_1)$ n'est plus colinéaire aux axes de la grille, et $|p_2 - p_1|$ est en fait légèrement supérieur à 1. On espère toutefois que cette modélisation, si elle interdit toute estimation exacte des constantes numériques, ne modifiera pas l'asymptotique de l'erreur. En supposant R suffisamment grand pour que les points de la courbe restent toujours bien proche du cercle de rayon R, on peut donc approcher notre trajectoire L par la courbe affine par morceaux L_P, reliant les points p_n, avec :

$$p_0 \quad = \quad R + 0\mathrm{i}, \tag{30}$$

et, pour tout n dans $[\![1, l]\!]$,

$$\begin{aligned} p_n &= \left(1 + \frac{\mathrm{i}}{R}\right) p_{n-1} && (31)\\ &= \left(1 + \frac{\mathrm{i}}{R}\right)^n p_0 && (32)\\ |p_n - p_{n-1}| &= 1 + o(1) && (33)\\ &\simeq 1. && (34) \end{aligned}$$

La courbe L_P relie donc $l+1$ points, pour être de longueur l. La courbe théorique L_T étant le cercle de centre 0 de rayon R, on peut approcher $\mathrm{Err}(L, L_T)$ comme suit :

$$\begin{aligned} \mathrm{Err}(L, L_T) &\simeq \mathrm{Err}(L_P, L_T) && (35)\\ &\simeq \frac{1}{l+1} \sum_{n=0}^{l} |p_n| - R && (36)\\ &= \frac{1}{l+1} \sum_{n=0}^{l} R\left(\left|1 + \frac{\mathrm{i}}{R}\right|^n - 1\right) && (37)\\ &= \frac{R}{l+1} \sum_{n=0}^{l} \left(\mathrm{e}^{n \log\left(1+\frac{1}{R^2}\right)/2} - 1\right) && (38)\\ &= \frac{R}{l+1} \sum_{n=0}^{l} \left(\frac{n}{2R^2} + o\left(\frac{n}{2R^2}\right)\right) && (39)\\ &\simeq \frac{l}{4R} && (40)\\ &\propto \frac{l}{R}. && (41) \end{aligned}$$

L'erreur moyenne est donc proportionnelle à l (les dérives successives s'accumulent), et inversement proportionnel au rayon de courbure R_C. Doubler la résolution d'une image de référence (passer de Lena 256px × 256px à Lena 512px × 512px par exemple), tout en gardant une longueur "physique" des lignes constante divisera par deux l'erreur apparente : l et R_C sont tous deux doublés par le changement d'échelle, et l'erreur $\mathrm{Err}(L, L_T)$ reste donc la même... Mais, le côté d'un pixel ayant été "divisé par deux", le résultat s'en trouve bien être deux fois plus satisfaisant.

Gradient de norme constante, calculé Nous souhaitons maintenant apprendre à calculer le gradient sur la grille de façon à se rapprocher au plus près de cette erreur "irréductible" en $1/R_C$, qui ne saurait être éliminée qu'en améliorant nos schémas d'interpolation et d'intégration du gradient, ou en améliorant la résolution de notre image. Là encore, nous voudrions avoir à intégrer un champ $\nabla \tilde{I}(\rho, \theta) = \vec{u}_\theta$, mais sans tricher. Une image possédant un tel champ de gradient ne pouvant être réalisée, nous avons donc contourné le problème : pour chaque valeur du rayon de courbure R_C, et avec l une longueur de ligne fixée, on construit une image à traiter I_{R_C}, définie par :

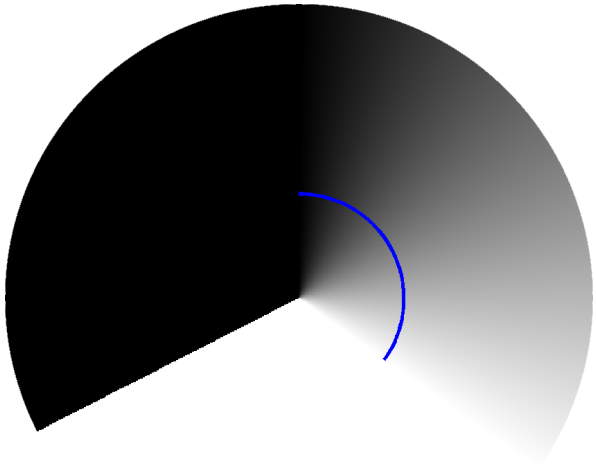


FIGURE 5 – L'image $I_{R_C=91}$, ainsi que la ligne de gradient au rayon de courbure constant associé, d'une longueur $l = 200$px.

$$I_{R_C}(\rho, \theta) = \begin{cases} 0 & \text{si } \theta < 0 \\ 255 \frac{R_C}{l} \theta & \text{si } 0 \leqslant \theta \leqslant \frac{l}{R_C} \\ 255 & \text{si } \frac{l}{R_C} < \theta \end{cases} \quad (42)$$

$$\nabla I_{R_C}(\rho, \theta) = \begin{cases} \vec{0} & \text{si } \theta < 0 \\ 255 \frac{R_C}{l\rho} \vec{u}_\theta & \text{si } 0 \leqslant \theta \leqslant \frac{l}{R_C} \\ \vec{0} & \text{si } \frac{l}{R_C} < \theta \end{cases} \quad (43)$$

Une ligne représentative de la déviation à rayon de courbure R_C fixé sera alors calculée, sur I_{R_C}, en partant du point $(\rho = R_C, \theta = 0)$, sur une longueur l : voir Figure 5. Nous pouvons maintenant évaluer

l'influence de la méthode de calcul du gradient, de son traitement et de son interpolation sur la qualité de nos lignes de champ.

Analyse des graphes de la Figure 15 Les graphes calculés sont cette fois-ci conformes à nos prévisions : Là où les lignes calulées sans traitement ne voient pas leur qualité augmenter avec le rayon de courbure, un simple filtrage gaussien de l'image suffit à retrouver des performances similaires au cas idéal du paragraphe précédent. Une mise en garde toutefois : un lissage trop fort (combinaison de méthodes) ou un filtrage médian, loin d'aider, induisent une perte d'information et une dégradation des courbes. Attention, donc, à ne pas trop dénaturer notre image par des traitements excessifs, un filtrage gaussien de l'image et du gradient (ce qui revient au même) semble bien suffisant.

Notons, pour finir, la très faible influence de la méthode d'interpolation du gradient sur le résultat final : c'est que nous nous trouvons ici en présence d'un courant fort et *très* régulier, comme illustré Figure 1b. Retenons donc que ce choix n'a de véritable importance qu'au voisinage des discontinuités et changements brusques de directions du champ.

3 Affichage des lignes

Paramètres d'études La précision du tracé à l'échelle sous-pixellique a été traitée dans la section précédente : il est désormais temps de nous intéresser à l'apparence globale de notre image. Une fois l'exactitude du traçé garantie, comment obtenir une visualisation qui soit à la fois facile à lire et riche en informations ?

Lissage de l'image et du gradient Comme on l'a vu plus haut, une régularisation du champ à intégrer semble bien nécessaire. La Figure 6a, calculée sans débruitage, ne contient que des lignes courtes ; en comparaison, les lignes macroscopiques des Figures 6b-6c-6d sont bien plus satisfaisantes, car calculées sur des images lissées.

Une régularisation du gradient est tout aussi pertinente : en réduisant le nombre de points d'annulations, elle accroît la longueur des courbes intégrées, et par là le confort de lecture. Les images de la Figure 7 nous invitent à privilégier un filtre médian pour l'image, et un filtre gaussien pour le gradient.

(a) Image naturelle, bruitée.
(b) Après un filtrage gaussien.
(c) Après un filtrage median.
(d) Après un filtrage de Wiener.

FIGURE 6 – Influence du niveau de bruit de l'image sur la longueur des lignes calculées.

Points de départ Quelles lignes doit-on choisir de tracer ? L'option la plus simple est de répartir nos points de départ sur une grille, ou de manière aléatoire en respectant une certaine distance minimale : c'est ce qui est fait sur les Figures 8a-8b.

De manière plus fine, on peut aussi privilégier une approche "topologique" du problème : les points de départ sont alors répartis au voisinage des extremas locaux de l'image, et donc tout autour des points critiques du champ. On obtient alors les cartes rayonnantes de la Figure 8c, auxquelles les cours de mathématiques nous ont habitué.

Cette dernière méthode, si séduisante qu'elle soit quand elle est appliquée à des champs de vecteurs issus de la physique (électromagnétisme, mécanique des fluides), n'est toutefois pas d'un grand intérêt pour nos images naturelles, à la "topologie" finale-

(a) Image naturelle, gradient moyenné par un filtre gaussien.

(b) Image naturelle, gradient lissé par un filtre median.

(c) Image après filtrage médian, gradient moyenné par un filtre gaussien.

(d) Image et gradient lissés par un filtre median.

FIGURE 7 – Influence du lissage de l'image et du gradient sur la longueur des lignes.

(a) Les points initiaux sont espacés régulièrement.

(b) Ici, de manière aléatoire.

(c) Ici, autour des extremas de l'image.

(d) En blanc, les zones d'intérêt.

FIGURE 8 – Influence du choix des lignes sur la lisibilité de l'image.

ment assez peu exploitable par un simple tracé de flot.

Longueur des lignes Limiter arbitrairement la longueur des lignes à tracer n'est guère satisfaisant : une succession de vermicelles ne rend pas vraiment compte de la réalité continue du champ, et la Figure 9b reste bien plus lisible que la Figure 9a. Attention toutefois à ne pas tomber dans l'excès inverse : une ligne de champ ne pouvant a priori s'arrêter qu'en un point d'annulation du gradient (relativement rares sur une image débruitée), notre image peut finir par ressembler à un sac de noeuds, comme dans la Figure 10a...

Pour contourner cette difficulté, on peut utiliser la transparence comme dans la Figure 10b, le recouvrement des lignes de champs correspondant à une intensité plus importante. Une autre option consiste à empêcher les lignes de s'approcher les unes des autres à moins de, par exemple, 10px : nous utilisons pour cela un système de masque, mis en application dans les Figure 9c-9d.

Densité des lignes Pour un affichage avec transparence, pas de doute : plus il y a de lignes, mieux c'est. Mais sinon, que faire ? Un écart interlignes du même ordre de grandeur que la taille caractéristique des détails de l'image nous semble raisonnable : rajouter plus de lignes ne ferait qu'alourdir la visualisation – voir Figure 11.

Épaisseur et couleur Le tracé de nos lignes décidé, on peut encore utiliser le *style* du trait pour ajouter des informations. C'est ce que nous avons choisi de faire dans la Figure 11c, utilisant une coloration périodique de nos courbes pour en indiquer le sens d'évolution, et modulant l'épaisseur du trait en fonction de la norme du gradient sous-jacent.

Évaluation critique Deux modes de visualisation nous semblent combiner densité d'information et clarté. Le premier, c'est l'utilisation de la

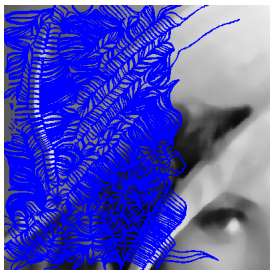
(a) Lignes longues d'au plus 50px.

(b) Lignes longues d'au plus 1000px.

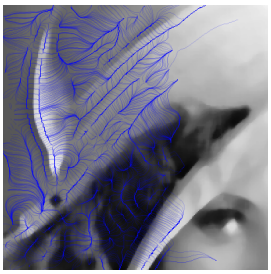
(c) Lignes espacées d'au moins 3px.

(d) Lignes espacées d'au moins 9px, plus courtes sur le chapeau.

FIGURE 9 – Influence de la longueur des lignes sur la lisibilité de l'image.



(a) Sans contrainte de longueur.

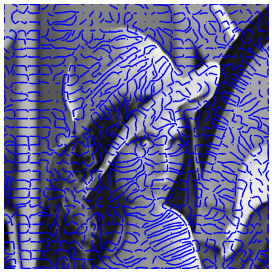


(b) Sans contrainte de longueur, mais avec transparence.

FIGURE 10 – Utilisation de la transparence pour permettre la superposition des lignes.

transparence : les images obtenues sont riches et immédiatement compréhensibles. Après tout, elles mettent en valeur l'interprétation physique des lignes de gradient, le ruissellement des eaux selon la plus forte pente, comme dans la Figure 12. Mais cette élégance a un coût très important, car les lignes, en plus d'être calculées en de très nombreux points, le sont sans critère d'arrêt – on entend bien que les trajectoires se recoupent. On pourrait bien sûr paralléliser les calculs de lignes, utiliser un GPU, mais cela dépasserait de loin le seuil de nos compétences...

(a) Densité trop faible.



(b) Densité trop forte.

(c) Densité satisfaisante.

FIGURE 11 – La densité du réseau de lignes affecte la lisibilité de l'image.

Un deuxième mode de visualisation, plus économique, nous est heureusement accessible : avec une densité de lignes bien choisie, l'allure du champ de gradient est tout à fait manifeste, comme sur la Figure 11c.

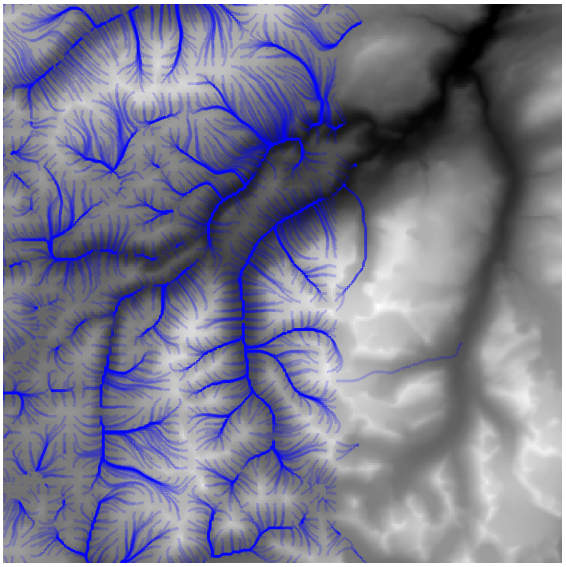


FIGURE 12 – L'écoulement des eaux dans le fjord de Jotunheimen...

4 Conclusions et perspectives

Nous savons maintenant calculer et afficher de manière précise et claire les lignes de gradient d'une image. Ce problème, intéressant en soi, mêlait le calcul d'un champ de vecteur intrinsèquement continu à partir de données 2D discrètes à la visualisation de son flot sur une image toute entière.

Terminons donc ce rapport en donnant quelques applications possibles de notre travail.

Usage pédagogique Nous avons déjà cité plus haut la simulation grossière de l'écoulement des eaux à partir d'un relevé toppgraphique, Figure 12, qui permet d'illustrer simplement la notion de gradient d'une force conservative. La flexibilité du procédé permettant de tourner les vecteurs avant d'intégrer le champ, on peut aussi réaliser très simplement de petites animations illustrant le passage "continu" des lignes de gradients (pas de rotation) aux lignes de niveaux (un quart de tour).

Nous tournant vers la physique, on peut enfin remarquer l'adéquation entre notre problème et la dynamique d'un système Hamiltonien conservatif à un degré de liberté. Rappelons simplement que si $H(q,p)$ est le Hamiltonien du système, avec q la position et p le moment, les équations du mouvement s'écrivent :

$$\dot{q} = \frac{\partial H}{\partial p}, \qquad \dot{p} = -\frac{\partial H}{\partial q}, \tag{44}$$

soit :

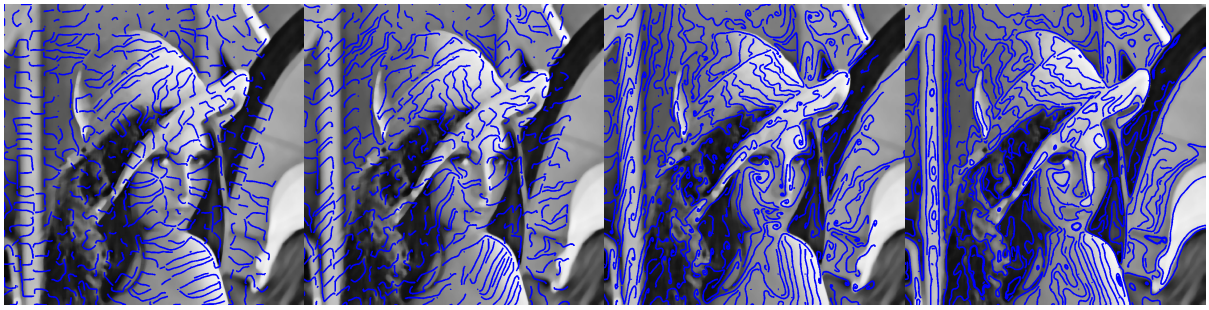
$$\begin{pmatrix} \dot{q} \\ \dot{p} \end{pmatrix} = R_{-90^\circ}(\nabla H(q,p)). \tag{45}$$

Aussi, avec $x = q$, $y = p$, en prenant pour image notre Hamiltonien $H(p,q)$ (échantillonné sur la grille), et en tournant notre champ de gradient d'un quart de tour dans le sens des aiguilles d'une montre avant de l'intégrer, les lignes obtenues correspondent aux trajectoires de notre mobile dans l'espace des phases.

Calcul des lignes de niveau d'une image Cette intégration des gradients "tournés" nous offre, par la même occasion, un nouvel algorithme de calcul des lignes de niveau d'une image : il serait sans aucun doute pertinent d'en comparer les performances avec celles du module llview.

Limites de nos visualisations Dans la littérature, d'autres procédés de représentation des champs de vecteurs existent : on peut notamment penser à la Line Integral Convolution (voir [1]), simple et efficace, qui peut éventuellement être colorée. En comparaison, notre méthode d'intégration des streamlines semble bien compliquée, mais offre néanmoins une très grande flexibilité.

Pistes pour des travaux futurs Une achitecture logicielle simple et robuste étant maintenant bien établie, une exploration exhaustive des différents paramètres du problème semble à portée. Un effort ultérieur devrait sans aucun doute se porter sur la phase d'interpolation et de régularisation du gradient : de nombreuses méthodes sont présentées dans la littérature (utilisation de noyaux reproduisants, d'équations de diffusion ou de moyennes le long des streamlines, interpolation par splines, ...), et nous n'avons fait ici qu'entrevoir la richesse du problème.



(a) $\theta = 0°$ (b) $\theta = 45°$ (c) $\theta = 79°$ (d) $\theta = 90°$

FIGURE 13 – Évolution des lignes lorsque le champ de gradient est tourné d'un angle θ avant intégration.

Références

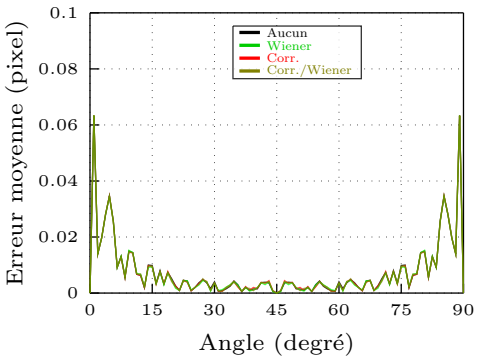
[1] Brian Cabral and Leith Casey Leedom. Imaging vector fields using line integral convolution. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pages 263–270, New York, NY, USA, 1993. ACM.

[2] Udo Diewald, Tobias Preußer, and Martin Rumpf. Anisotropic diffusion in vector field visualization on euclidean domains and surfaces. *IEEE Trans. Vis. Comput. Graph.*, 6(2) :139–149, 2000.

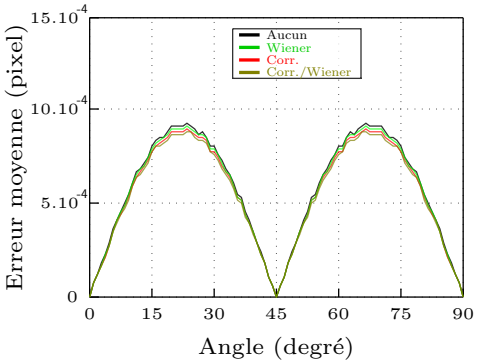
[3] Thomas Ertl, Thomas Klein, Katrin Bidmon, and Andreas Hub. Vector field visualization. Notre de cours, Université de Stuttgart, 2003.

[4] Bruno Jobard. *Visualisation de champs de vecteurs bidimensionnels à base de streamlines*. PhD thesis, Dunkerque, Grenoble, 2000. Th. : informatique.

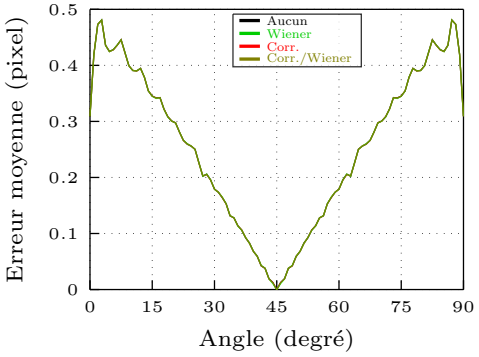
[5] Henry Kang, Seungyong Lee, and Charles K. Chui. Coherent line drawing. In *ACM Symposium on Non-Photorealistic Animation and Rendering (NPAR)*, pages 43–50, August 2007.



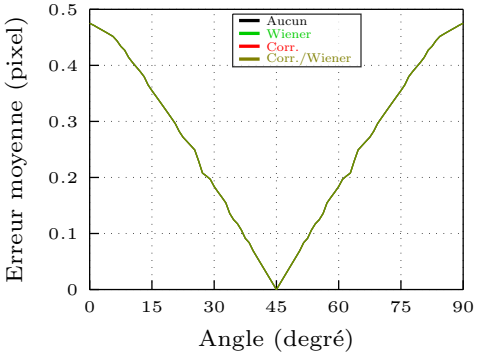
(a) Gradient constant par morceaux, sans traitement.



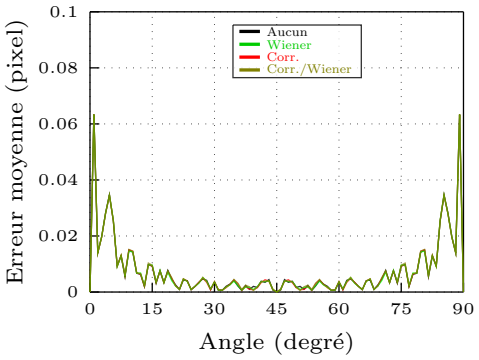
(b) Interpolation bilinéaire du gradient, sans traitement.



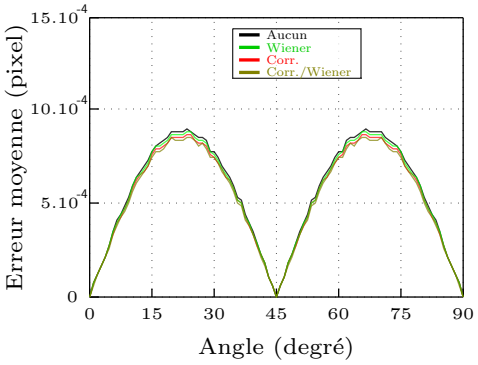
(c) Gradient constant par morceaux, puis filtrage médian.



(d) Interpolation bilinéaire du gradient, puis filtrage médian.

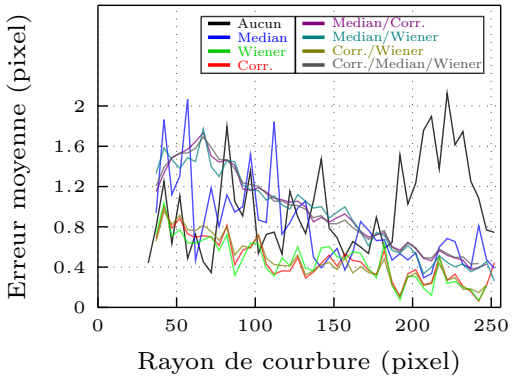


(e) Gradient constant par morceaux, puis filtrage gaussien.

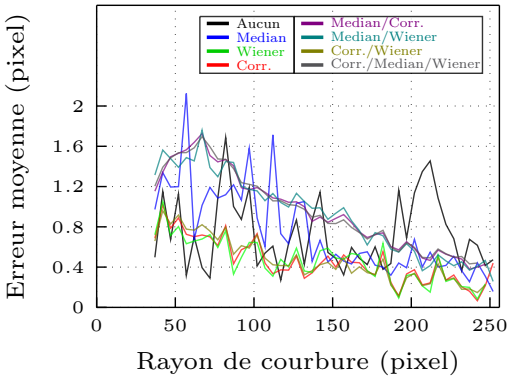


(f) Interpolation bilinéaire du gradient, puis filtrage gaussien.

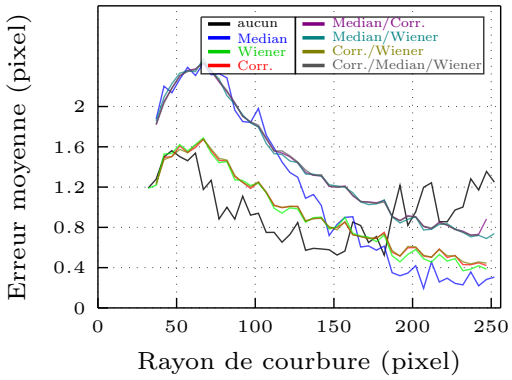
FIGURE 14 – Influence du traitement de l'image sur la justesse des lignes de champ, calculées sur l'image radiale avec une longueur de 100px en partant d'un cercle de 50px de rayon. Dans tous les cas, on utilise la méthode de Runge-Kutta à l'ordre 4-5, coûteuse mais précise. La plupart des traitements de l'image sont sans grand effet ; notons toutefois les performances désastreuses du filtrage médian appliqué à l'image, avec des erreurs moyennes de l'ordre du pixel – les courbes ne sont ici pas tracées.



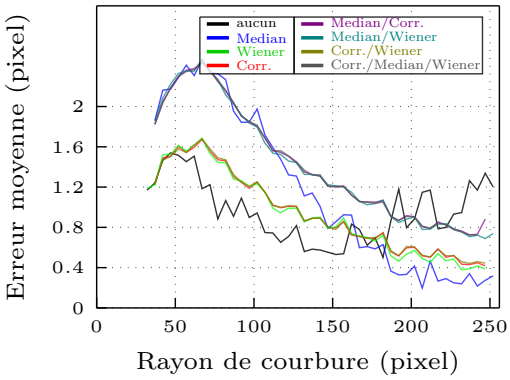
(a) Gradient constant par morceaux, sans traitement.



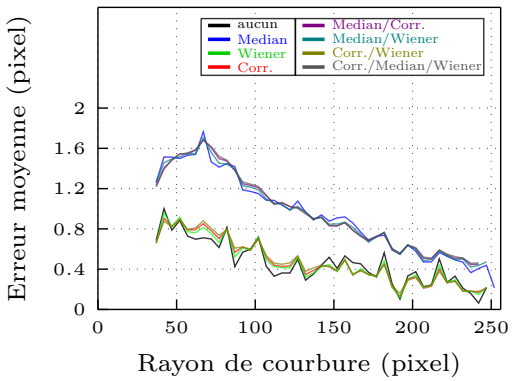
(b) Interpolation bilinéaire du gradient, sans traitement.



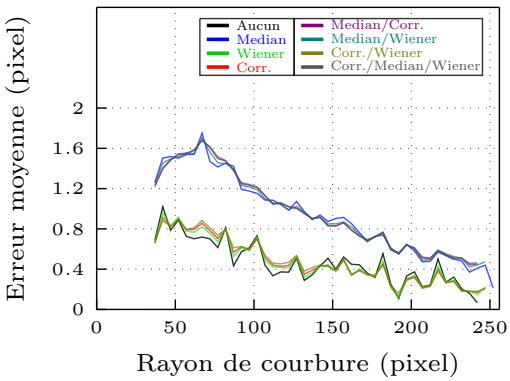
(c) Gradient constant par morceaux, puis filtrage médian.



(d) Interpolation bilinéaire du gradient, puis filtrage médian.



(e) Gradient constant par morceaux, puis filtrage gaussien.



(f) Interpolation bilinéaire du gradient, puis filtrage gaussien.

FIGURE 15 – Influence du traitement de l'image sur la justesse des lignes de champ, calculées sur les images "à rayon de courbure constant" I_{Rc}. Dans tous les cas, on utilise la méthode de Runge-Kutta à l'ordre 4-5, coûteuse mais précise.