
Université Grenoble Alpes
INP PHELMA
Physique, Électronique, Matériaux
Signal Communication Multimédia

Université Grenoble Alpes
INP ENSE3
École Nationale Supérieure de l'Eau,
de l'Énergie et de l'Environnement
Signal Communication Multimédia

Demosaicing

Rapport de projet Image Analysis

26/01/2024

par

VOUILLON Antoine

Encadrants universitaires :
DALLA MURA Mauro
MULLER Matthieu

Introduction

Ce rapport a pour but de rendre compte du projet de Demosaicing dans le cadre du cours d'Image analysis. Pour expliquer le demosaicing attardons nous sur la manière dont une image est physiquement enregistré par une pellicule. La lumière est captée par une pellicule, mais pour avoir l'information des différentes couleurs, on dispose des filtres de couleur devant cette même pellicule. Cela créer donc un pavage qui peut être de différente forme (comme par exemple le pavage bayer ou quad-Bayer).

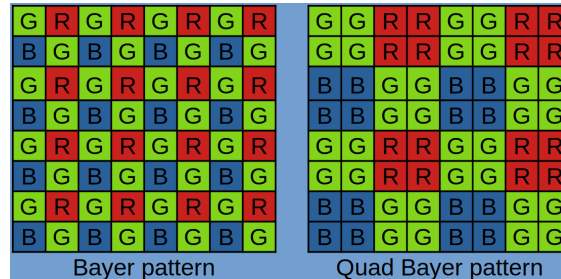


FIGURE 1 – Pattern des filtres de couleurs les plus courants

Cependant cette méthode d'aquisition à un défaut. Pour chaque canal l'image n'est pas complète. Il faut donc pour chaque canal (Rouge, vert et bleu) compléter les informations des pixels manquants.

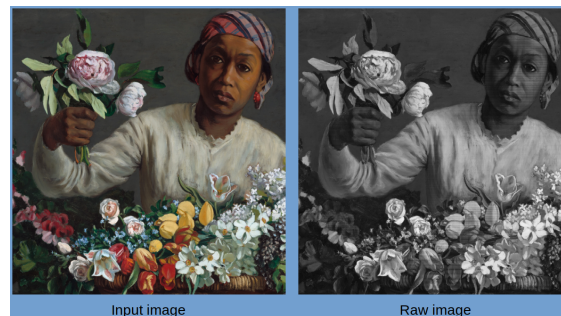


FIGURE 2 – Exemple d'image à reconstruire

Cette procédure s'appelle donc le démosaicing. Dans ce rapport nous allons voir les différentes techniques qui existent afin de réaliser cette opération, puis nous vérifierons l'efficacité de ces techniques.

Pour cela nous utiliserons des images provenant d'une banque d'images de la National Galery of Art. Nous comparerons les différentes techniques sur 4 images en particulier.

1 Méthodes de Demosaicing

Dans un premier temps nous allons décrire les différentes méthodes que nous allons tester et comparer dans ce rapport.

1.1 Bi-linear interpolation

Il existe beaucoup de type d'interpolation différente comme par exemple celle des "plus proches voisins" ou enncore "l'interpolation bicubique". Ici nous avons décidés de nous limiter à l'interpolation la plus couramment utilisée en imagerie numérique, "l'interpolation biliéaire". Il s'agit, à partir d'une grille régulière de pixels, de déterminer la fonction bi-linéaire permettant de relier quatre points adjacents de la grille.

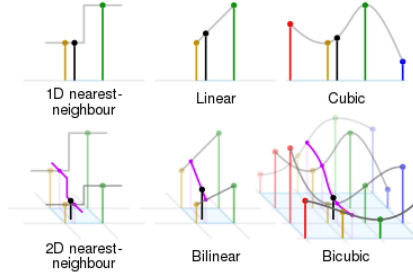


FIGURE 3 – Exemples d'interpolation 1D et 2D

Cette méthode était déjà implémentée dans le projet. Ca sera donc l'image étalon à laquelle on pourra comparer les autres méthodes.

1.2 Algorithme de Kimmel

L'algorithme se décompose en trois étapes :

1. Interpolation de la couleur verte
2. Interpolation des autres couleurs à l'aide de la couleur verte
3. Phase de correction

Intéressons nous d'abord aux deux premières phases de l'algorithme sans la phase de correction.

1.2.1 Interpolation du vert

Chaque pixel vert non déterminé est entouré de quatre pixels verts déterminés lors de l'acquisition. On calcule donc un pixel à partir des quatre pixels qui les entourent.

$$\begin{array}{|c|c|c|} \hline & G_2 & \\ \hline G_4 & G_5 & G_6 \\ \hline & G_8 & \\ \hline \end{array} \quad G_5 = \frac{E_2 G_2 + E_4 G_4 + E_6 G_6 + E_8 G_8}{E_2 + E_4 + E_6 + E_8}$$

E_i – weight function

$$D_{xd}(G_5) = \frac{G_3 - G_7}{2\sqrt{2}} \quad (3)$$

$$D_{yd}(G_5) = \frac{G_1 - G_9}{2\sqrt{2}} \quad (4)$$

Les poids sont ensuite calculés avec la forme suivante :

$$E_i = \frac{1}{\sqrt{1 + D^2(G_5) + D^2(G_i)}} \quad (5)$$

FIGURE 4 – Notations des pixels voisins et calcul du pixel central

Les poids E_i sont calculés de la manière suivante. Tout d'abord on calcule les gradients directionnel :

$$D_x(G_5) = \frac{G_4 - G_6}{2} \quad (1)$$


$$D_y(G_5) = \frac{G_2 - G_8}{2} \quad (2)$$

Avec DG_i la dérivée directionnelle en direction de G_i .

En appliquant cet procédé à tout les points non déterminés de la grille on complète l'image sur le canal vert.

1.2.2 Interpolation du bleu et du rouge

L'interpolation des deux couleurs suivante est très similaire. Prenons par exemple le canal rouge. Dans un premier temps on cherche à constituer une grille semblable à la grille verte initiale. Pour cela on sélectionne les pixels de la manière suivante et on calcul la valeur du pixel central de la manière suivante :



$$R_5 = G_5 \frac{E_1 \frac{R_1}{G_1} + E_3 \frac{R_3}{G_3} + E_7 \frac{R_7}{G_7} + E_9 \frac{R_9}{G_9}}{E_1 + E_3 + E_7 + E_9}$$

E_i – weight function

FIGURE 5 – Notations des pixels voisins et calcul du pixel central

Cette méthode se base sur le constat qu'au sein d'un même objet, les rapports entre vert et rouge restent les mêmes.

Une fois le pixel central déterminé on calcul le reste des pixels manquants avec la même méthode que l'on a utilisé pour les pixels verts.

On répète la même opération pour les pixels bleus et on obtient une image complètement interpolée avec les bonnes dimensions.

1.3 Correction

Afin de respecter la supposition initiale, on cherche à égaliser le rapport entre toutes les couleurs. Pour cela on passe à une étape de correction. A chaque itération de cette correction on met à jour les valeurs des pixels vert de la manière suivante :

$$G_5^B = B_5 \frac{E_2 \frac{G_2}{B_2} + E_4 \frac{G_4}{B_4} + E_6 \frac{G_6}{B_6} + E_8 \frac{G_8}{B_8}}{E_2 + E_4 + E_6 + E_8},$$

$$G_5^R = R_5 \frac{E_2 \frac{G_2}{R_2} + E_4 \frac{G_4}{R_4} + E_6 \frac{G_6}{R_6} + E_8 \frac{G_8}{R_8}}{E_2 + E_4 + E_6 + E_8},$$

$$G_5 = \frac{G_5^R + G_5^B}{2}$$

FIGURE 6 – Mise à jour de G_i

Ensuite on met à jour les pixels verts et bleus de la manière suivante :

$$B_5 = G_5 \frac{\sum E_i \frac{B_i}{G_i}}{\sum E_i}, i \neq 5, \quad R_5 = G_5 \frac{\sum E_i \frac{R_i}{G_i}}{\sum E_i}, i \neq 5$$

FIGURE 7 – Mise à jour de R_i et B_i

Afin de faire converger les valeurs on répète cette opération trois fois.

2 Implémentation et Résultats

2.1 Metriques

Pour comparer les différentes méthodes, nous utiliserons deux mesures différentes. La première est le rapport signal-bruit de crête, et est calculée de la manière suivante :

$$\text{PSNR}(x, y) = \frac{10 \log_{10} [\max(\max(x), \max(y))]^2}{|x - y|^2}$$

Cette mesure est souvent utilisée pour comparer des images reconstruites ou dégradées à l'image originale, ce qui est le cas ici. Plus le PSNR est élevé, mieux c'est.

La deuxième mesure est davantage basée sur les statistiques de l'image et se calcule comme suit :

$$\mathbf{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1) + (2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (6)$$

L'indice de similarité structurelle (SSIM) est une mesure utilisée pour quantifier la similarité entre deux images en évaluant leur luminance, leur contraste et leurs informations structurelles. Il fournit un score numérique indiquant la similarité visuelle, les valeurs SSIM les plus élevées suggérant une ressemblance plus étroite entre les images.

2.2 Résultats



FIGURE 8 – Image 1



FIGURE 10 – Image 3

Method	PSNR	SSIM
bilineat interpolation	34.63	0.9502
Algo de Kimmel sans correction	35.55	0.9574
Algo de Kimmel avec correction	4.45	0.2978

Method	PSNR	SSIM
bilineat interpolation	31.98	0.8941
Algo de Kimmel sans correction	32.79	0.9046
Algo de Kimmel avec correction	5.04	0.2504



FIGURE 9 – Image 2



FIGURE 11 – Image 4

Method	PSNR	SSIM
bilineat interpolation	30.31	0.8430
Algo de Kimmel sans correction	31.44	0.8602
Algo de Kimmel avec correction	7.92	0.4377

Method	PSNR	SSIM
bilineat interpolation	29.88	0.8145
Algo de Kimmel sans correction	30.95	0.8352
Algo de Kimmel avec correction	9.69	0.4586

2.3 Analyse

On voit que sur toute les images sont bien améliorer avec la méthode d'interpolation bilinéaire, Mais qu'à chaque fois la méthode de Kimmel sans correction est légèrement supérieure. Cependant la méthode de Kimmel avec correction dis-fonctionne totalement. En effet j'ai remarqué que lorsque la phase de correction est terminée, certaine valeurs deviennent aberrante et dépasse 1 avec des 10aine d'ordres de grandeur. Je n'a malheureusement pas réussi à régler ce problème.

J'ai donc décidé d'implémenter la méthode de Kimmel sans phase de correction dans ma fonction de reconstruction.

3 Conclusion

On a réussi à améliorer la qualité de la reconstruction des images à partir de la méthode de Kimmel. Cependant je n'ai pas réussi à faire marcher la phase de correction qui aurait encore améliorer les résultats. Malheureusement je n'ai pas réussi à trouver d'où venait le problème d'explosion de certaines valeurs. De plus je n'ai pas eu le temps de m'intéresser à d'autres pattern comme le pattern Kodak ce qui aurait été également intéressant. Voilà les points d'amélioration que je vois pour aller plus loin avec la méthode de Kimmel.

- [1] *"An Improved Demosaicing Algorithm" by Alexey Lukin, Denis Kubasov*
<http://ultra.sdk.free.fr/docs/Image-Processing/Demosaic/An%20Improved%20Demosaicing%20Algorithm.pdf>