

Implémentation d'article de recherche
Removing Camera Shake via Weighted Fourier
Burst Accumulation

Sofiane Horache, Roman Fenioux

April 2017

Introduction

Le problème adressé ici est le flou de bougé, présent notamment dans les photographies amateur prises à la main dans des environnements mal éclairés. Pour avoir une photographie suffisamment exposée il est souvent nécessaire d'augmenter le temps d'exposition lorsqu'on ne peut ou ne veut pas agir sur l'ouverture ou la sensibilité de l'appareil. Mais si la position ou l'orientation de l'appareil change durant l'acquisition, on voit apparaître du flou dans la direction du mouvement.

Approche de l'article

Le mouvement de l'appareil durant la prise de vue est modélisé par une convolution de l'image par un noyau de flou. La direction du mouvement est aléatoire d'une image à l'autre et on veut donc récupérer l'information que contient chaque image pour obtenir une image nette. L'approche utilisée ici consiste à prendre une rafale d'images au lieu d'une seule, puis à les combiner pour éliminer ce flou. Cette technique permet même de débruiter l'image.

Pour une séquence de M images on a donc :

$$v_i = u_i * k_i + n_i$$

Pour ce faire l'article propose de calculer une moyenne pondérée de leurs transformées de Fourier pour éviter d'avoir à faire une estimation explicite du noyau de flou.

$$u_p(x) = \mathcal{F}^{-1} \left(\sum_{i=1}^M w_i(\zeta) \cdot \hat{v}_i(\zeta) \right) (x)$$
$$w_i(\zeta) = \frac{|\hat{v}_i(\zeta)|^p}{\sum_{j=1}^M |\hat{v}_j(\zeta)|^p}$$

Le paramètre p est un entier qui contrôle la manière dont sont agrégés les résultats. Si $p = 0$ la somme se ramène à une moyenne des transformées de Fourier des différentes images. Lorsque $p \rightarrow \infty$ cela revient à prendre pour chaque fréquence la maximale qu'elle atteint dans une des images.

Notre implémentation

Nous avons travaillé en C++ avec la librairie OpenCV.

Nous avons d'abord implémenté la méthode de l'article sur des images superposées bien recalées, puis nous avons implémenté l'algorithme ORSA(Optimized RANSAC) pour le recalage des images. L'avantage de cette méthode est que contrairement à RANSAC, il n'y a aucun paramètre à fixer. L'algorithme est automatique. Le principe est de trouver pour chaque homographie, le Nombre de Fausse Alarme(NFA) minimum. Le NFA est définie :

$$\text{NFA}(k) = (n - N_{sample}) \binom{n}{k} \binom{k}{N_{sample}} (\epsilon_k^2 \alpha_0)^{k - N_{sample}}$$

où n est le nombre de match, k est le nombre d'inlier ϵ_k est la kieme plus petite erreur. et α_0 est la probabilité d'avoir une erreur de moins de un pixel. Nous avons pris $N_{sample} = 4$. Voici le pseudo-code du recalage:

```

input : 2 images non recalés, niter
Calcul du SIFT sur les deux images;
Recherche des Matches avec l'algorithme Flann(plus proche voisin);
bestNFA = ∞;
inliers = ∅;
bestH = I;
for i allant de 1 à niter do
    tirer  $N_{sample}$  matches de manières aléatoire;
    calcul de l'homographie H;
    if l'homographie n'est pas dégénérée(bien conditionné) then
        calcul des erreurs ( $\epsilon_k$ ) pour chaque matcha;
        on trie les ( $\epsilon_k$ );
        on calcule  $NFA(k*) = \min_k(NFA(k))$ ;
        inliers = les  $k*$  premiers inliers;
        if bestNFA plus grand que  $NFA(k*)$  then
            bestNFA= $NFA(k*)$ ;
            bestH = H;
    output: bestH
end
```

^autilisation de la distance symétrique. pour un match(x_i, x'_i) et H une homographie, on calcule l'erreur e $e = d(Hx_i, x'_i) + d(H^{-1}x'_i, x_i)$

On peut ensuite recaler les images sur elles même. Ensuite, on peut appliquer le défloutage.

input : Images recalées avec ORSA

Nos résultats

Images déjà recalées

Nous avons d'abord testé la méthode sur des images déjà recalées : les images anthropologies utilisées dans l'articles ainsi que d'autres que nous avons faites nous même avec des noyaux de flous correspondant à un mouvement linéaire. Dans la figure 1 (page 3) on peut comparer le résultat obtenu à l'image originale utilisée et à la meilleure des images floues.

Nous avons ajouté du bruit gaussien uniforme sur les images pour vérifier la robustesse de la méthode et comme annoncé par l'article cela ne dégrade pas le résultat. On observe même comme attendu un effet de débruitage plus ou moins efficace selon la quantité d'images utilisées.

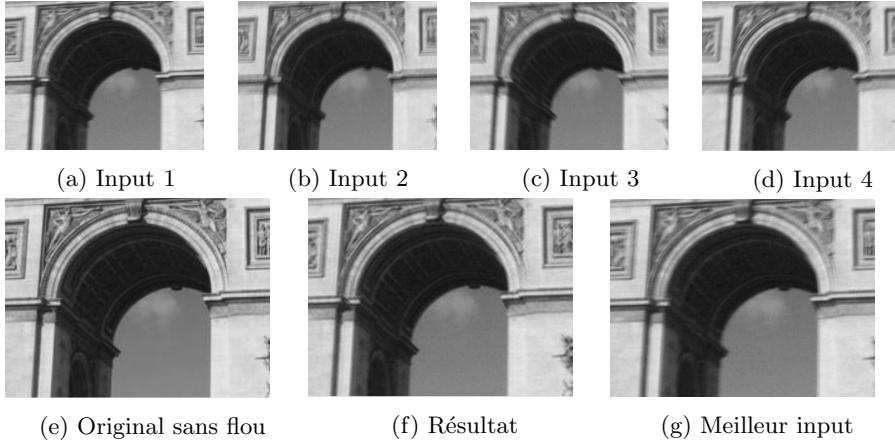


Figure 1: Résultat de l'algorithme pour 4 images recalées

Influence des paramètres

Le paramètre p permet de doser — A COMPLETER — voir figure 2 page 4.

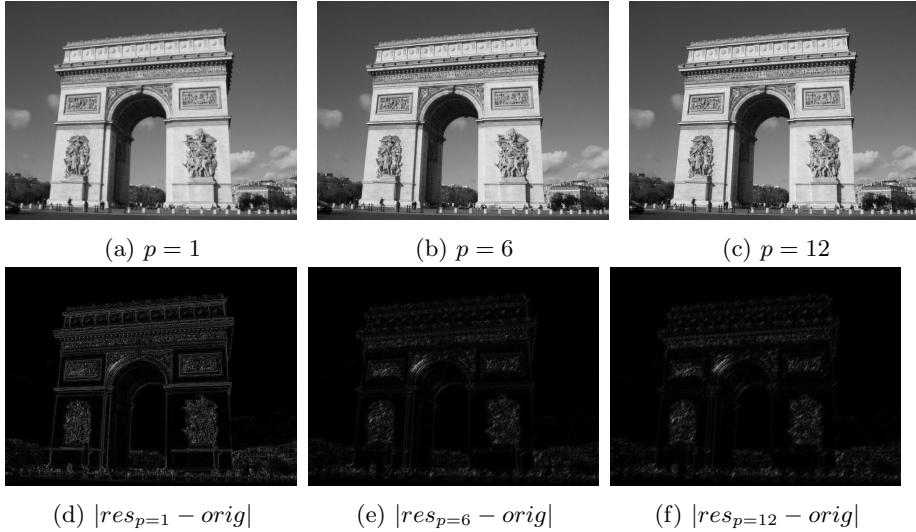


Figure 2: Influence du paramètre p

Images non recalées

Mettre une image d'orsa avec inliers

Dans la phase de recalage, le nombre d'itération utilisé joue un rôle très important. Comme on le voit sur la figure 3 (page 5), en doublant le nombre d'itérations on obtient un recalage plus fin qui permet un meilleur résultat. En effet la moindre erreur de recalage (quelques pixels) se traduit par des artefacts

dus à des effets de déphasage (Figure 3c page 5).

Faire un test woods avec recalage.

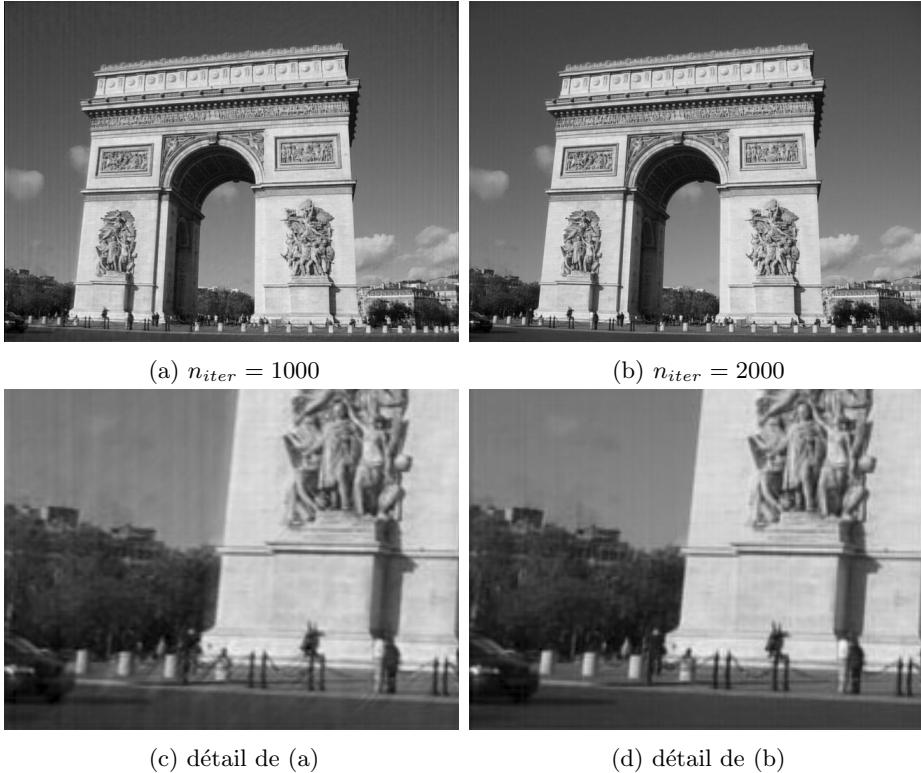


Figure 3: Influence du nombre d’itérations

Analyse et Commentaires

La méthode proposée par l’article donne de bons résultats dans le cas d’images bien recalées. Elle est très robuste au bruit et à des noyaux de flous relativement grands (jusqu’à 5 pixels), pas nécessairement en mouvement rectiligne. Cependant il y a quelques points qui peuvent poser problème.

Recalage des images

Le recalage s’avère être le point critique de notre implémentation. Une erreur lors du recalage d’une seule des images engendre de très gros défauts dans le résultat comme on le voit sur la figure 4 (page 5). Ce problème nous a conduit à rejeter les images dont l’estimation d’homographie s’appuie sur un nombre de points trop faibles. Nous utilisons donc ici le critère : image rejetée si $n_{inliers} < 100$. Nous sommes bien sûr conscient qu’un meilleur critère devrait être utilisé.



Figure 4: Résultat suite à un mauvais recalage

La phase de recalage peut prendre beaucoup de temps, mais est extrêmement importante pour avoir de bons résultats. Sur nos exemples il est nécessaire d'avoir au moins 2000 iterations dans l'algorithme ORSA pour obtenir un recalage satisfaisant. L'article évite cette question en supposant que le recalage est fait immédiatement grâce aux gyroscopes des smartphones, mais cela peut constituer une limite à l'application de la méthode en temps réel dans d'autres appareils. Cependant on peut imaginer que ce problème se règle avec l'augmentation des capacités de calcul des appareils.

Flou non homogène : mouvement rotatif

Une limite intrinsèque à cette méthode est l'hypothèse que le flou peut être modélisé comme une convolution par un noyau. C'est le cas pour les mouvements de translation et de certaines rotations mais ce n'est plus possible lorsque l'appareil subit une rotation autour de son axe optique. La PSF dépend alors de la position du pixel dans l'image, ce qui ne peut pas être modélisé par une convolution avec un noyau.

Ce cas de figure peut pourtant se produire en situation réelle comme nous l'avons constaté (voir Figure 5, page 5). Le résultat est très bon au centre de l'image mais se dégrade vers les bords. Une solution envisageable pourrait être d'appliquer l'algorithme localement sur des portions de l'image suffisamment petites pour que l'hypothèse de convolution par un noyau de flou soit valable.



(a) Résultat



(b) Centre de l'image



(c) Bas de l'image

Figure 5: Résultat suite à un flou de rotation