

Lecture et déchiffrement d'une photo de peinture

Théo CLAYETTE – Thibaut ETIENNE – Arnaud SOULIER
Université de Montpellier

6 Mars 2018

1 Introduction

Dans le but de déchiffrer une peinture imprimée et préalablement chiffrée par un algorithme de mélange pseudo-aléatoire, il faut avant tout faire des prétraitements sur l'image pour obtenir une image contenant uniquement la représentation de la peinture et non le fond qui l'entoure.

Les prétraitements choisis et présentés dans ce rapport sont dans l'ordre :

- la transformation en niveaux de gris
- la transformation en image binaire (noir et blanc)
- la détection des angles de la peinture sur l'image
- la transformation affine de la peinture

Une fois toutes ces opérations effectuées, on peut appliquer le déchiffrement sur la peinture uniquement et obtenir la véritable œuvre. La figure 1 présente la photo utilisée pour tester les algorithmes présentés dans ce rapport.



FIGURE 1 – Photo utilisée pour les test de lecture et de déchiffrement d'une peinture imprimée

2 Niveaux de gris

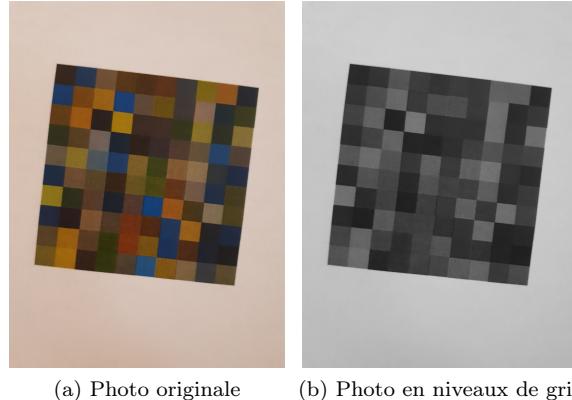
La première étape de ce processus de traitement est la transformation de l'image couleur (format .ppm) en niveaux de gris (format .pgm). Pour ce faire, il suffit d'appliquer la formule suivante (cf. équation 1) à tous les pixels de l'image.

$$data[i] = 0.299 \cdot R[i] + 0.587 \cdot G[i] + 0.114 \cdot B[i] \quad (1)$$

avec

- $data[i]$ la valeur en niveaux de gris du pixel i de l'image
- $R[i]$ la valeur de la composante rouge du pixel i de l'image
- $G[i]$ la valeur de la composante verte du pixel i de l'image
- $B[i]$ la valeur de la composante bleue du pixel i de l'image

Il en résulte donc une image de même dimension mais composée uniquement de nuance de gris (cf. figure 2(b)).



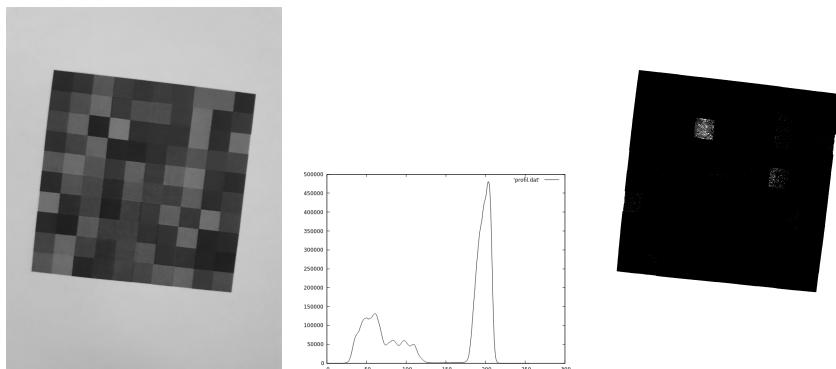
(a) Photo originale (b) Photo en niveaux de gris

FIGURE 2 – Conversion en niveaux de gris de l'image

3 Image binaire

À partir de l'image en niveaux de gris, on passe à une image binaire (noir et blanc) avec un simple seuillage des valeurs. Pour chaque pixel de l'image, si sa valeur est supérieure au seuil choisi, alors cette valeur devient 255 (blanc). Si sa valeur est inférieure au seuil choisi, alors cette valeur devient 0 (noir).

La figure 3(c) montre le résultat de la conversion binaire de l'image en niveaux de gris.



(a) Photo en niveaux de gris (b) Histogramme de la photo en niveaux de gris (c) Photo binaire (seuil 120)

FIGURE 3 – Conversion en binaire de l'image

Pour obtenir la valeur de seuillage à utiliser pour faire cette conversion, il faut observer l'histogramme des valeurs de l'image en niveaux de gris et choisir une valeur cohérente (séparation nette entre les pixels les plus foncés et les plus clairs). On peut également automatiser ce choix en prenant par exemple la première valeur telle que 40% des pixels ont une valeur inférieure à ce choix.

4 Détection des angles

À partir de l'image binaire, la détection des angles est grandement facilité grâce aux faibles nombres de valeur possible (noir ou blanc) et la répartition séparée dans l'image.

L'algorithme utilisé ici part à tour de rôle des angles de l'image et parcours tous les pixels dans une zone entre cet angle et le centre de l'image à la recherche du pixel noir le plus proche de l'angle. Ce pixel sera alors l'angle de la peinture sur l'image. L'algo 1 représente l'itération d'un des angles. Il doit donc être reproduit et adapté trois fois pour retourner les quatre angles.

Algorithm 1 Trouve l'angle supérieur gauche

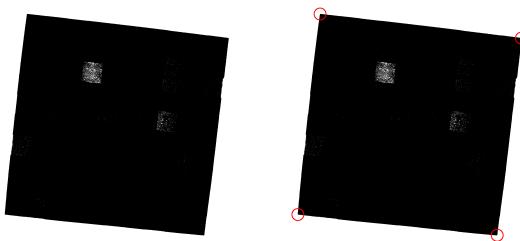
```

 $x \leftarrow height/2$ 
 $y \leftarrow width/2$ 
for  $i$   $0..height/2$  do
    for  $j$   $0..width/2$  do
        if  $data[i * width + j] = 0$  then
            if  $i + j < x + y$  then
                 $x \leftarrow i$ 
                 $y \leftarrow j$ 
            end if
        end if
    end for
end for

```

Les quatre formules pour calculer le point le plus proche de l'angle sont :

- $i + j < x + y$ pour l'angle supérieur gauche
- $j - i > y - x$ pour l'angle supérieur droit
- $i + j > x + y$ pour l'angle inférieur droit
- $i - j > x - y$ pour l'angle inférieur gauche



(a) Photo binaire (seuil 120) (b) Photo binaire avec angles retournés par l'algorithme

FIGURE 4 – Détection des angles de la peinture sur l'image binaire

Pour que cet algorithme retourne les bonnes coordonnées pour chaque angle,

il faut que l'image binaire ne présente aucun point noir en dehors de la zone centrale (la peinture). Pour empêcher cela, il faut soit choisir une valeur plus petite pour le seuillage de l'image (au risque de perdre des morceaux à l'intérieur ou sur les bords de la zone centrale), soit effectuer une ou plusieurs ouvertures (une érosion suivie d'une dilatation) pour supprimer les points isolés dans le fond de l'image sans toucher à la zone centrale.

5 Transformation affine

Une fois que les angles de la peinture sont définis, il ne reste plus qu'une étape pour que l'image soit prête à être déchiffré. Il faut effectuer une transformation affine afin d'obtenir une image contenant uniquement la peinture positionnée correctement (les bords droits sans pente).

Pour effectuer cette transformation, cinq points sont définis.

- E : point coulissant le long du coté supérieur
- F : point coulissant le long du coté gauche
- G : point coulissant le long du coté inférieur
- H : point coulissant le long du coté droit
- K : point d'intersection des segments EG et FH

Chacun des points E, F, G et H évolue sur une découpe de n points d'un des côtés (n correspondant aux dimensions de la nouvelle image).

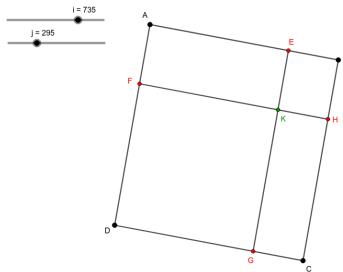


FIGURE 5 – Calcul des coordonnées pour la transformation affine

Chaque pixel $[i, j]$ de la nouvelle image (transformée) est défini par une copie de la valeur aux coordonnées du point K pour les i^{eme} points E/F et les j^{eme} points F/H (cf. figure 5).

6 Résultats finaux

La figure 6 montre la totalité du processus de lecture et de déchiffrement d'une peinture à partir d'une photo.

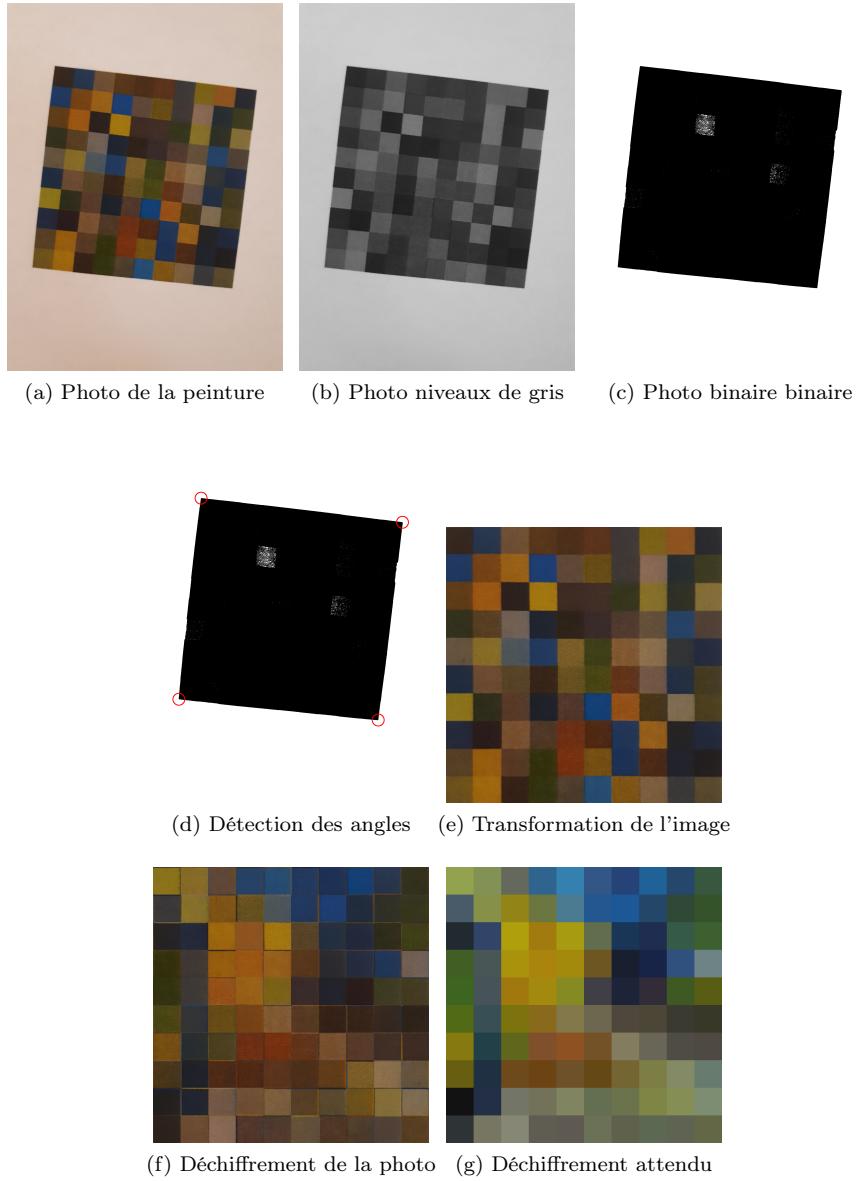


FIGURE 6 – Résultat du processus total de déchiffrement de la peinture à partir d'une photo