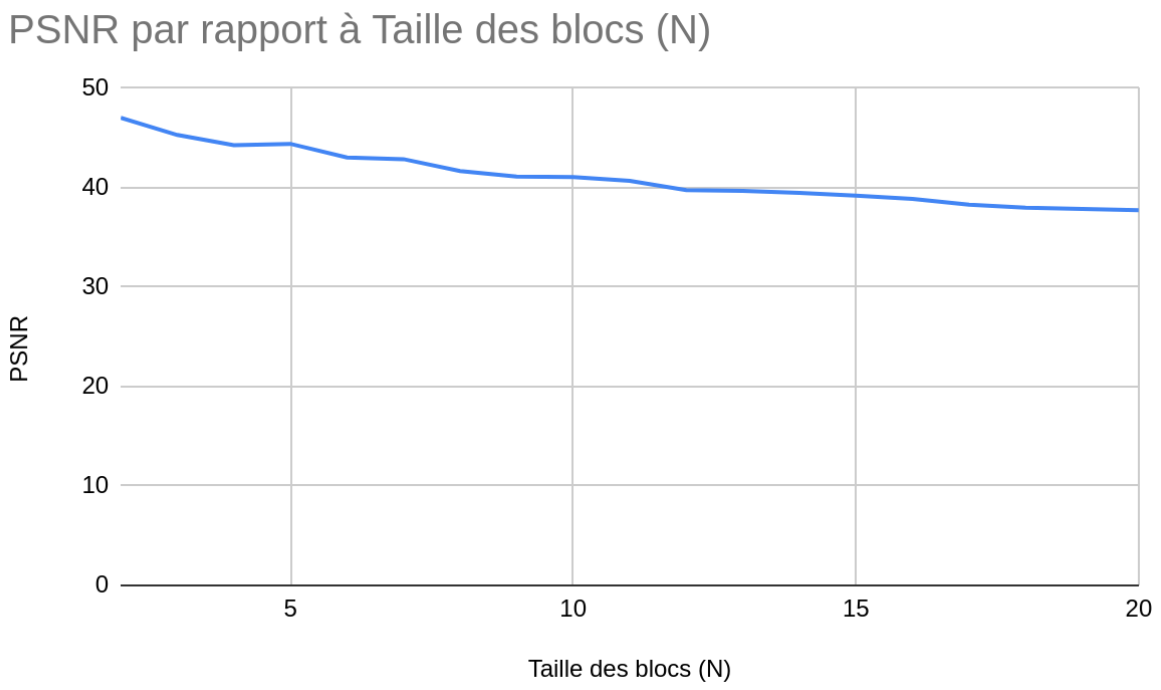
CR_4

Avancée

Une petite base de données de 7 photos de tableaux a été constituée afin de pouvoir tester notre algorithme.

Nous avons choisi une taille de bloc de 16*16 pixels car c'est le meilleur entre une qualité d'image déchiffrée correcte et une taille de bloc assez grande pour qu'elle soit identifiable sur une photo afin de délimiter les blocs.



Codage du chiffrement des images :

- Décomposition RGB de l'image
- Pour chaque composante : passage de l'image une image pour laquelle un pixel est la valeur moyenne d'un bloc de 16*16 → réduction de la taille de l'image
- Calcul de la suite de Chebyshev à partir de x_0 = 0.3 et k = 4.0 (la clé choisie 0.3 et 4.0)
- Tri des valeurs de cette suite
- Pour chaque composante, permutation des pixels selon le tri de la suite
- Regroupement des composantes

- ❖ Remise à l’échelle

Codage du déchiffrement des images (même principe que pour le chiffrement, mais en inversant les permutations)

→ ERREUR : MAUVAIS RÉSULTAT APRÈS DÉCHIFFREMENT (donc problème lors du chiffrement ou du déchiffrement)

On a inversé les étapes de recomposition à partir des 3 composantes et de remise à l’échelle, ce qui nous à permis d’obtenir un résultat.

On a donc effectué notre test sur l’image suivante :

Image de base

Les résultats ont été :

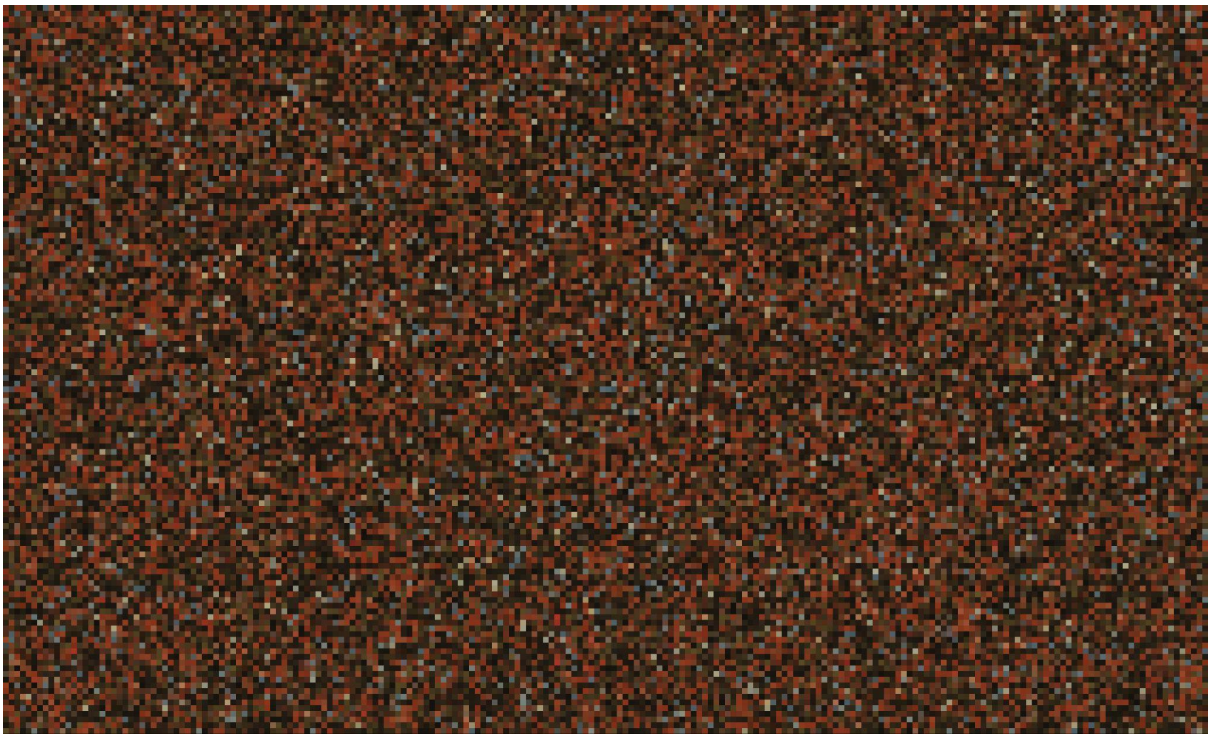


Image chiffrée

Image déchiffrée

Avec un PSNR entre l'image de base et l'image déchiffrée égale à 38.87dB, montrant que le déchiffrage est très correct.

Comme on peut le voir, le fait d'utiliser des blocs pour chiffrer l'image, amène une pixellisation de cette image, toutefois, nous craignons que si les blocs sont plus petits, il soit difficile de les retrouver à partir d'une photo.

Pour le recalage, on commence par appliquer un recalage sur une image sans rotation (créée par ordinateur à partir de notre image chiffrée).

L'image créée est la suivante :

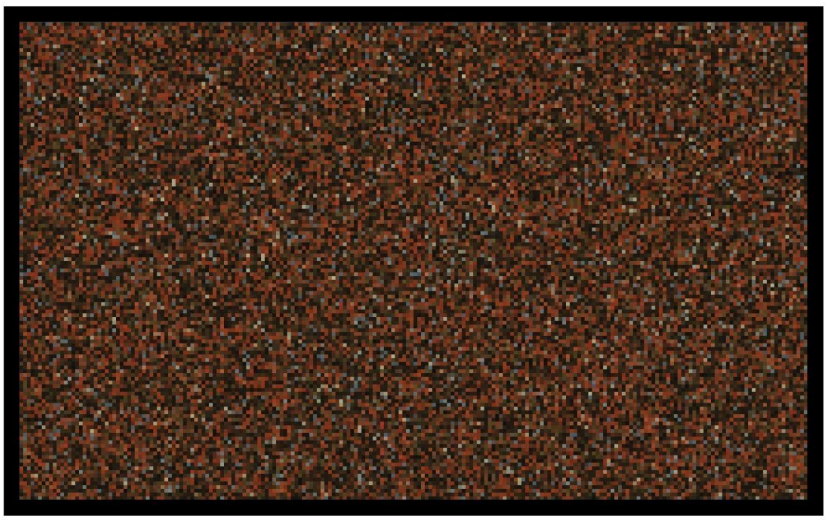


Image test représentant un tableau crypté entourré d'un cadre noir et posé sur un mur blanc

On identifie sur cette image le cadre de notre image chiffrée et on supprime le fond blanc sur l'image résultat.

Tableau crypté et son cadre isolé du fond

On identifie ensuite l'intérieur du cadre pour se débarrasser de celui-ci.

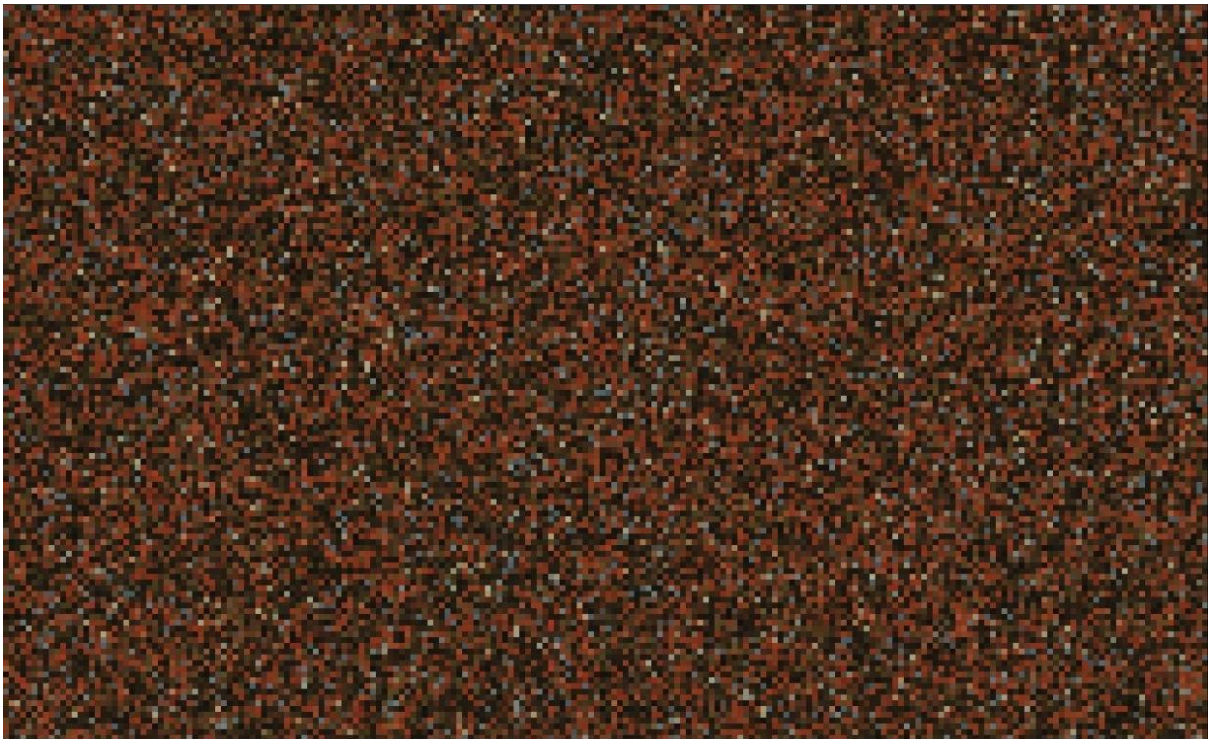


Tableau crypté après la suppression du cadre

Il nous reste maintenant à identifier la taille de nos blocs. Pour cela, un petit damier a été inséré dans le cadre (au coin en bas à droite). Il nous permet de retrouver plus facilement la taille de nos blocs.

Une fois la taille des blocs trouvée, il est très facile de re-découper notre image selon ces blocs.

Pour chaque bloc, on récupère le pixel central comme valeur du bloc (on peut aussi faire une moyenne pondérée par le centre), et on stocke ces pixels dans une nouvelle image qui nous suffit de redimensionner en bloc de 16*16 pixels pour ensuite pouvoir appliquer notre chiffrement.

Après une tentative effectuée, le résultat obtenu n'est pas trop mal, malheureusement, certaines lignes et colonnes fusionnent entre l'image chiffrée de base et l'image recalée, ce qui à pour conséquence de fausser le déchiffrement.
Il faut donc envisager soit de trouver une solution pour éviter que deux lignes/colonnes de l'image chiffrée devienne une dans l'image recalée soit, d'agrandir la taille des blocs (en perdant encore en qualité d'image déchiffrée) afin de limiter les cas où ce problème peut apparaître.