PROJET MAM3

Compression et traitement d'images par transformée de Fourier

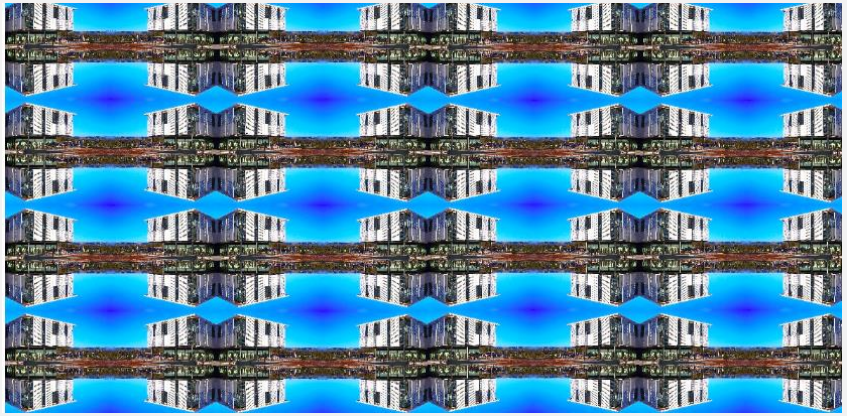
Veillon Camille, Ziad Zineb Keruzoret Matthieu, Piculescu Petru

SOMMAIRE

- Introduction
- DCT (Transformée de cosinus discrète) d'une image : Matrice de transition
- Algorithme : Initialisation, Compression, Décompression, Post-processing
- Résultats et interprétations
- Conclusion

INTRODUCTION

- **Objectif** : Utiliser des transformées de Fourier (élément clé du traitement du signal et des images) pour étudier et compresser l'information contenue dans une image, décomposer n'importe quelle image comme une combinaison linéaire de fonctions cosinus, en x et en y.



ALGORITHME INITIALISATION

- **Objectif :** Préparer l'image pour les étapes de compression et de décompression en normalisant les données et en respectant les contraintes nécessaires

1. **Chargement de l'image :**
    1. Utilisation de **imread** (matplotlib.pyplot) pour ouvrir l'image.
    2. Conversion en tableau **NumPy** avec des triplets RGB par pixel.
    3. On vérifie si l'image donnée est en format .jpg ou en format .png (différence de traitement pour les valeurs RGB).
2. **Ajustement des dimensions :**
    1. Troncature de l'image pour que les dimensions (x, y) soient des multiples de 8.
    2. Calcul via un modulo 8 sur les dimensions de la matrice.
3. **Centrage des valeurs :** Recentrage des valeurs RGB entre -128 et 127 pour chaque canal de couleur.
4. **Conversion finale :** Transformation des valeurs flottantes en entiers.

DCT D'UNE IMAGE

$$D_{k,l} = \frac{1}{4} C_k C_l \sum_{i=0}^{7} \sum_{j=0}^{7} M_{i,j} \cos\left(\frac{(2i+1)k\pi}{16}\right) \cos\left(\frac{(2j+1)l\pi}{16}\right),$$

Matrice de passage

$$D = PMP^T$$

Matrice de transition P

$$P_{k,n} = \sqrt{\frac{2}{N}} \cdot \begin{cases} \frac{1}{\sqrt{2}}, & si\ k = 0 \\ \cos\left(\frac{\pi(2n+1)k}{2N}\right), & si\ k > 0 \end{cases}$$

ALGORITHME
COMPRESSION AVEC MATRICE Q

- **Principe de la quantification :**

Conservation des modes les plus importants (basses fréquences).

Suppression des hautes fréquences (considérées comme du bruit).

- **Processus :**

Division terme à terme de la matrice **D** par une matrice de quantification **Q**.

Arrondi des valeurs obtenues à la partie entière.

- **Résultat :**

Une matrice très creuse (peu de valeurs non nulles)

- **Taux de compression** atteint : 85 à 90% grâce à la formule:

$$Q = \begin{pmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 13 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{pmatrix}$$

taux de compression =(1- $\frac{nombre\ de\ coefficients\ non\ nuls}{taille\ des\ matrices\ de\ couleurs * 3}$**) * 100**

ALGORITHME DÉCOMPRESSION AVEC MATRICE Q

- À partir de l'image compressée, les opérations inverses sont appliquées :

1. Découpage de l'image en blocs **8 × 8**.
2. Multiplication terme à terme de la matrice compressée par **Q :** obtention de l'image décompressée exprimée dans la base des cosinus.
3. Changement de base inverse pour obtenir l'image exprimée dans la base des intensités lumineuses
4. Réassemblage des blocs **8 × 8** pour reconstruire l'image originale.
5. Pour les images en couleur : la transformation est appliquée à chaque canal de couleur (rouge, vert, bleu).

ALGORITHME
COMPRESSION PAR FILTRAGE

- **But :** compresser une image et enlever le bruit
- **Principe :**

Troncature des informations au-delà d'une certaine fréquence.

Remplacement de l'étape de quantification (division/remultiplication par Q) par une troncature simple

Mise à 0 des coefficients **Di,k** de la matrice **D** vérifiant **i + k** ≥ **F**, où **F** est la fréquence de coupure (exemple : **F = 6**).

- **Avantages et Limites:**

Avantages : Plus rapide que la matrice Q de la norme JPEG.

Limites : Moins efficace en termes de compression et de qualité.

ALGORITHME POST-PROCESSING

- Pour évaluer la qualité de la compression, on compare :
- **La matrice originale** obtenue à l'ouverture du fichier de l'image.
- **La matrice résultante** après les étapes de compression et de décompression.
- **Calcul de l'erreur :**

Utilisation de la norme de la différence entre ces deux matrices pour mesurer la distance (erreur).

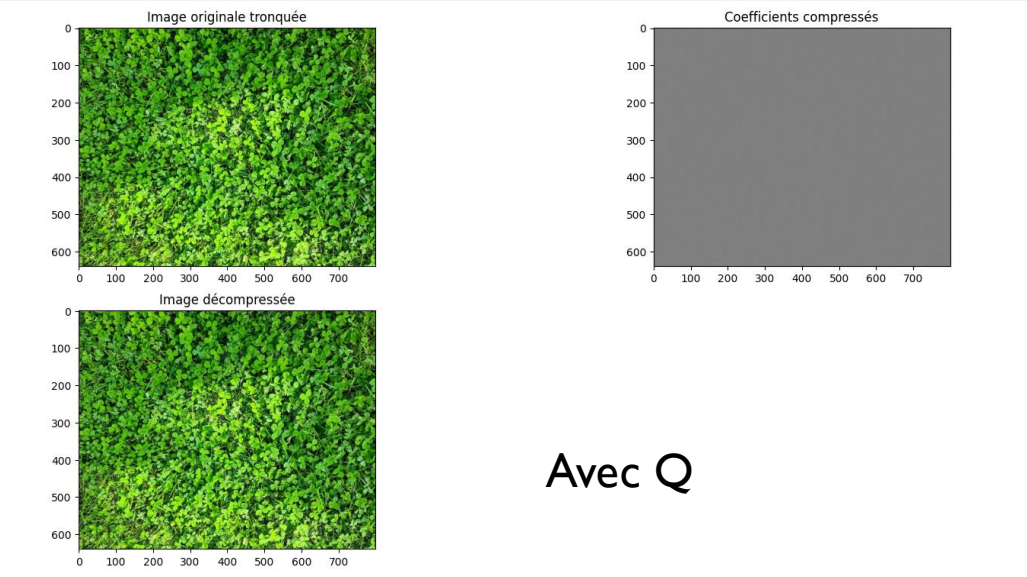
Le calcul est réalisé avec la fonction **numpy.linalg.norm** de la bibliothèque NumPy.

Ce pourcentage d'erreur permet d'évaluer l'impact de la compression sur la qualité de l'image.

Pour finir, nous pouvons sauvegarder les images obtenues grâce à la fonction **plt.imsave** pour sauvegarder en format .png et la fonction **.save** pour sauvegarder en format .jpg.

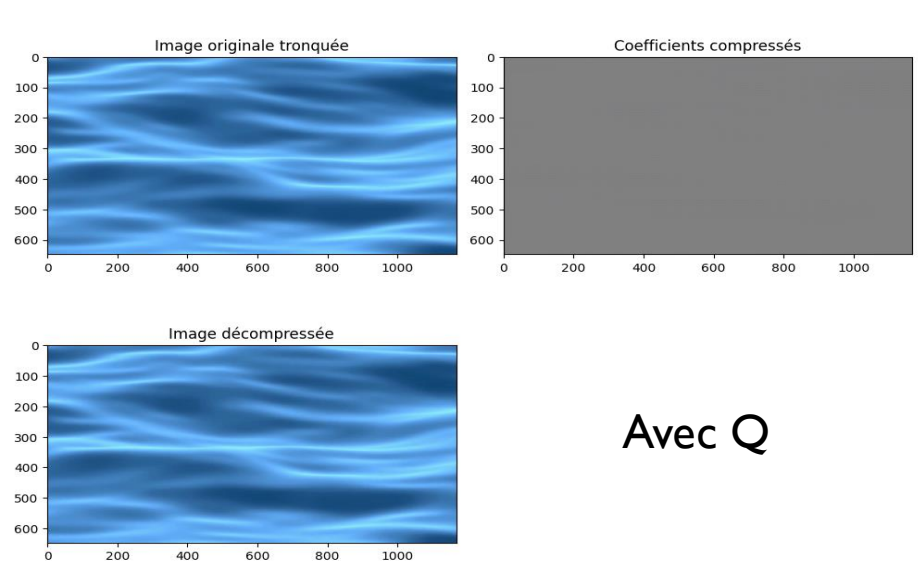
On pourra comparer les tailles de fichier afin de voir si la compression réduit la taille du fichier ou non.

RÉSULTATS ET INTERPRÉTATIONS



Avec Q

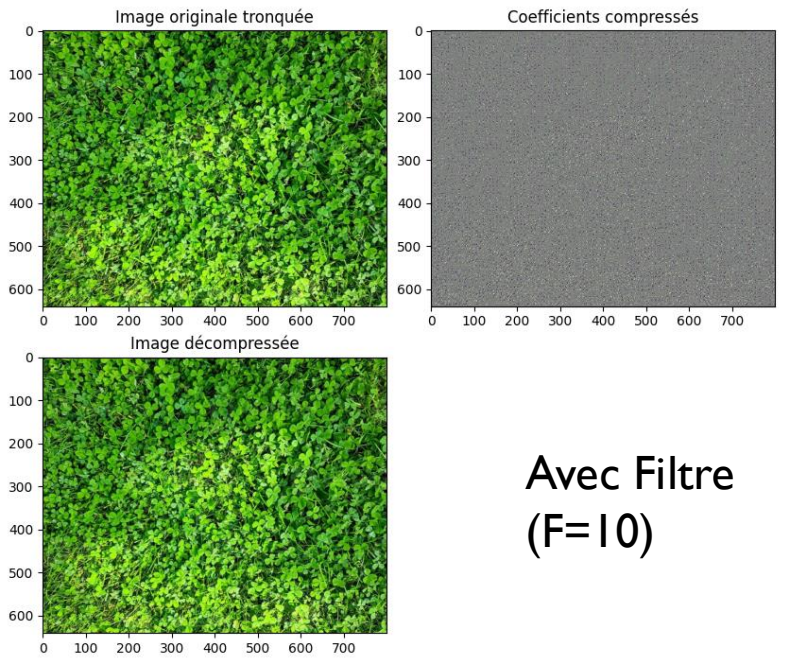
Taux de compression : 66%
Pourcentage d'erreur: 11,09%



Avec Q

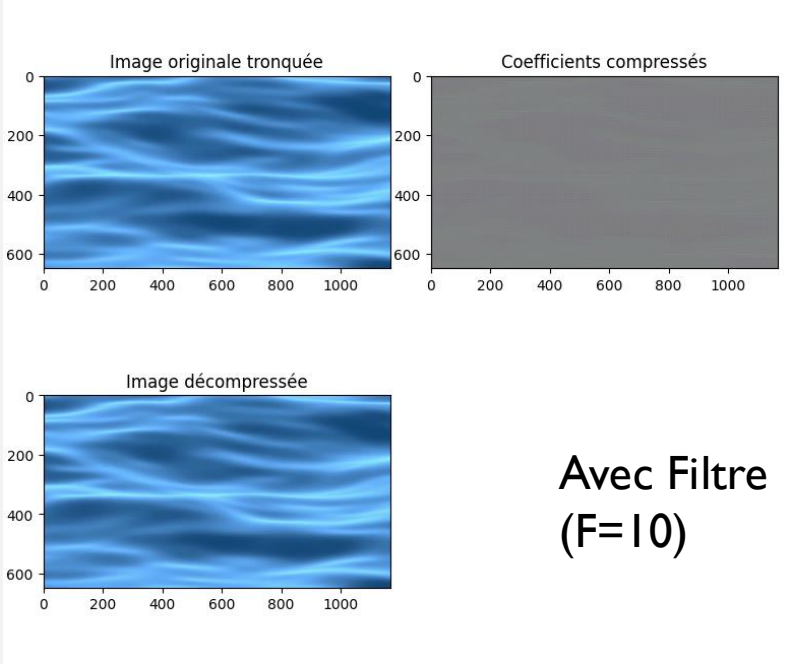
Taux de compression : 96%
Pourcentage d'erreur: 0,87%

RÉSULTATS ET INTERPRÉTATIONS



Avec Filtre
(F=10)

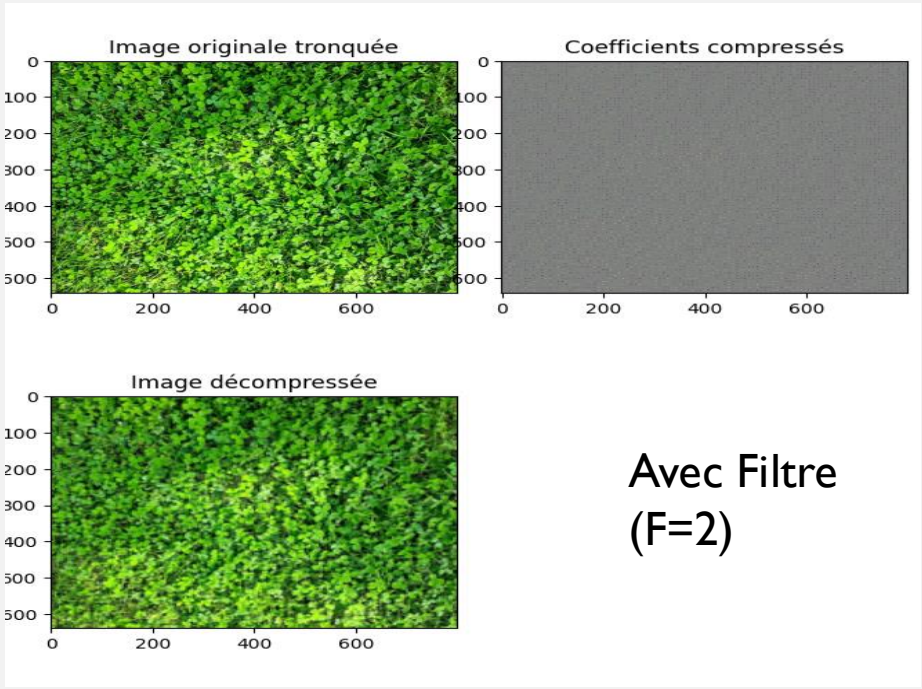
Taux de compression : 32%
Pourcentage d'erreur: 6,17%



Avec Filtre
(F=10)

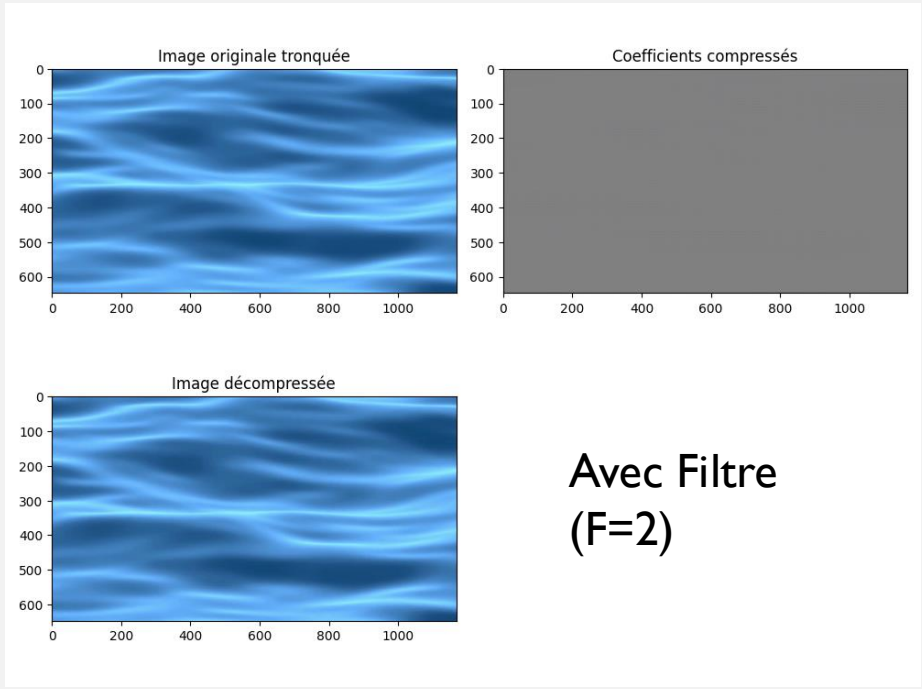
Taux de compression : 89%
Pourcentage d'erreur: 0,52%

RÉSULTATS ET INTERPRÉTATIONS



Avec Filtre
(F=2)

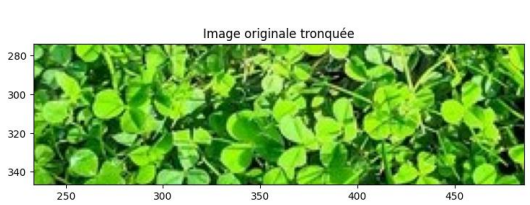
Taux de compression : 95%
Pourcentage d'erreur: 31,28%

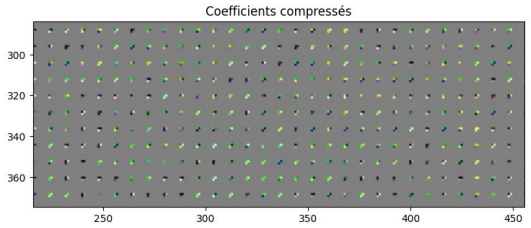


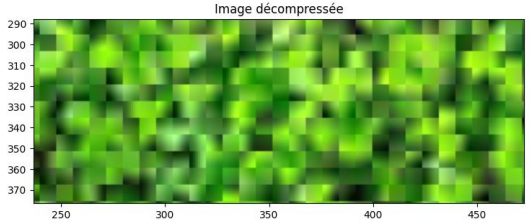
Avec Filtre
(F=2)

Taux de compression : 96%
Pourcentage d'erreur: 0,83%

RÉSULTATS ET INTERPRÉTATIONS

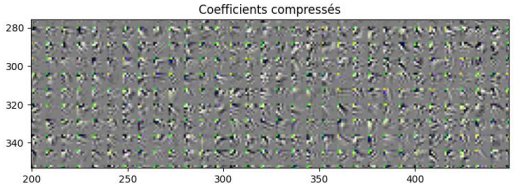






Avec Filtre
(F=2)

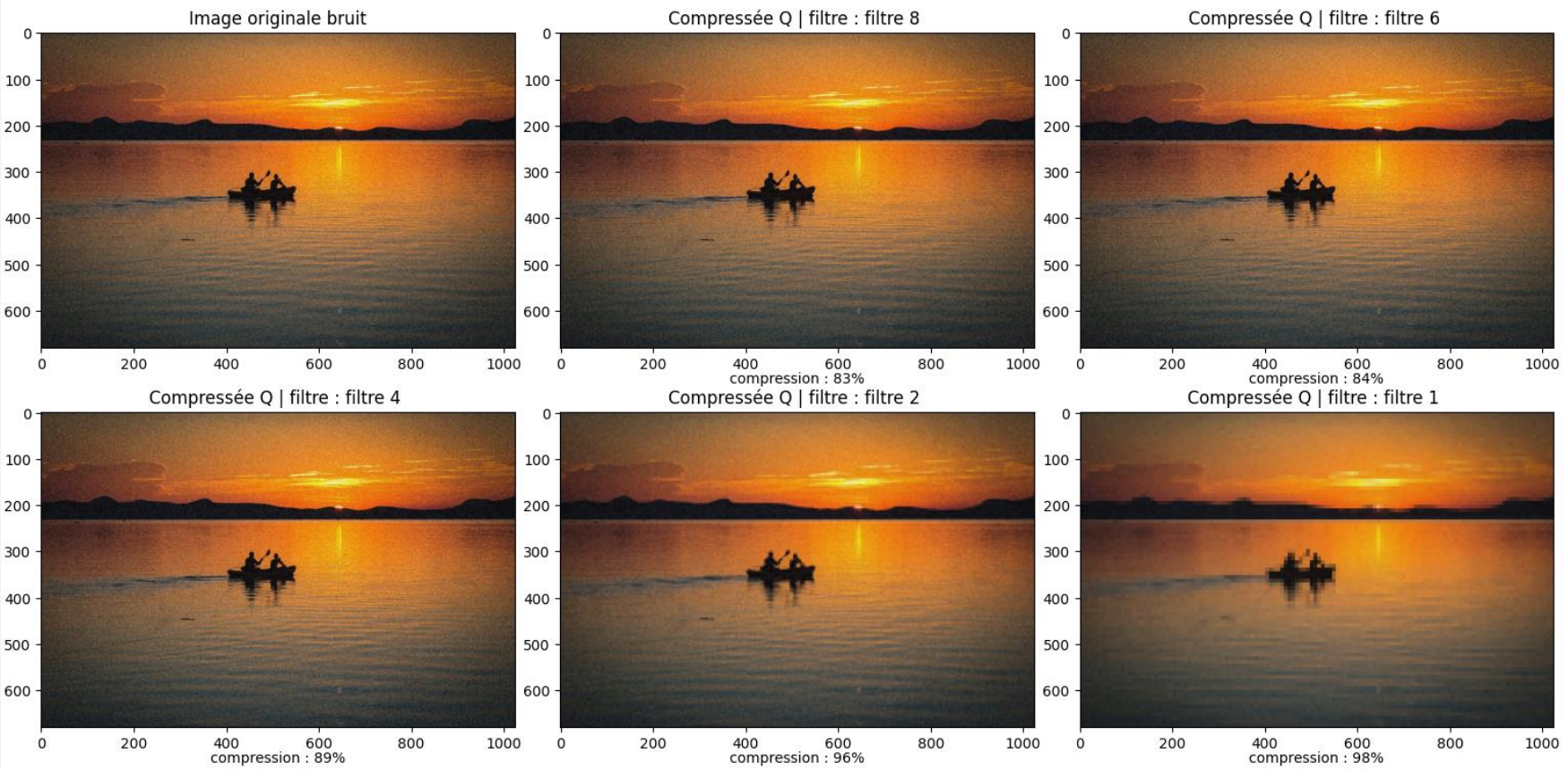
Taux de compression : 95%
Pourcentage d'erreur: 31,28%



Avec Filtre
(F=10)

Taux de compression : 32%
Pourcentage d'erreur: 6,17%

RÉSULTATS ET INTERPRÉTATIONS

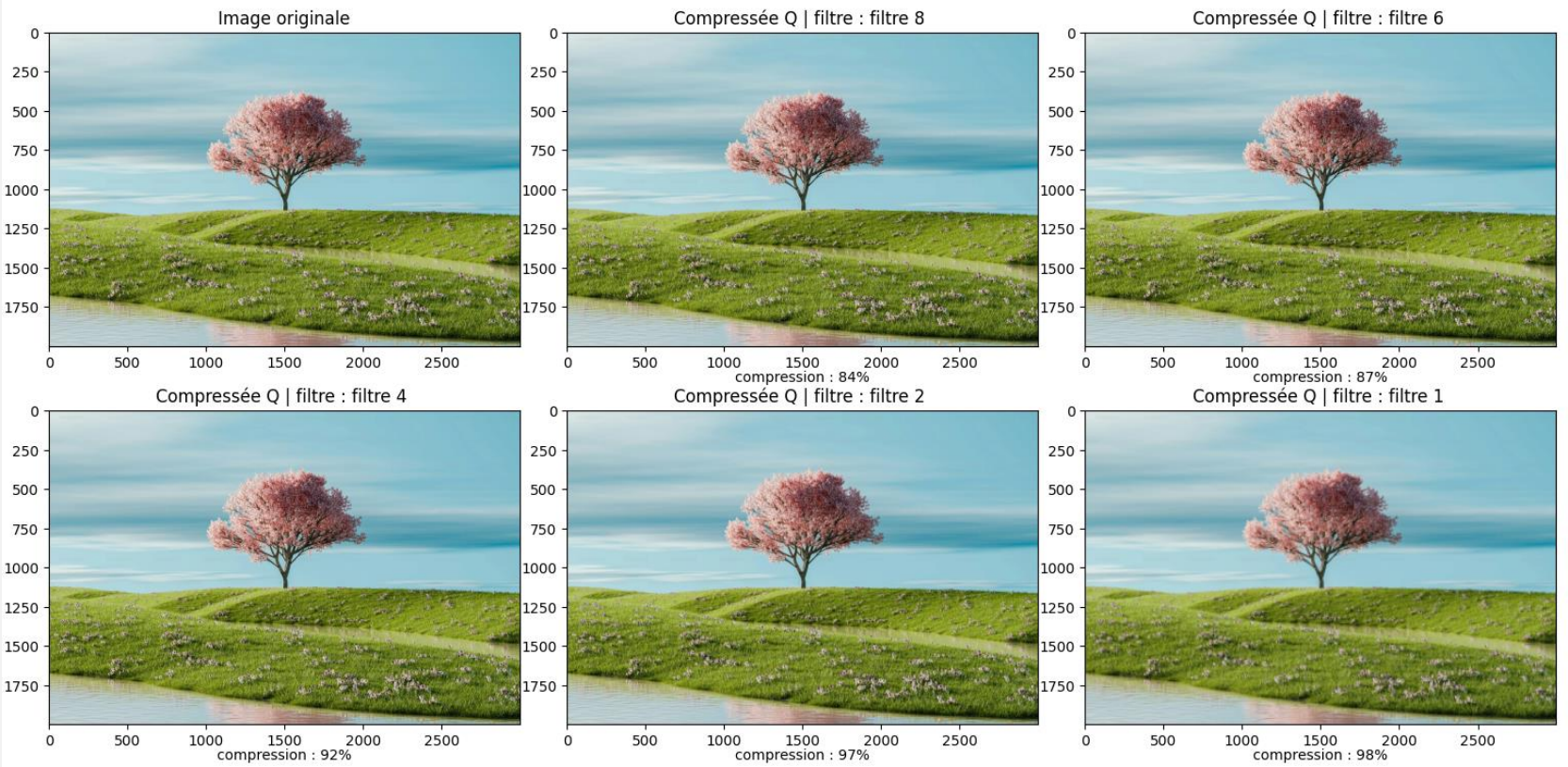


Matrice Q fixe, valeur du filtre F varie sur image bruitée

RÉSULTATS ET INTERPRÉTATIONS

Matrice Q varie, valeur du filtre F fixe

RÉSULTATS ET INTERPRÉTATIONS



Matrice Q fixe, valeur du filtre F varie

RÉSULTATS ET INTERPRÉTATIONS

CONCLUSION

- On peut en conclure que si on prend une matrice de quantification avec des valeurs très élevées et en prenant un filtre très bas, alors on pourrait obtenir une image très compressée.

- On peut s'imaginer également rajouter des détails sur une image de faible qualité, comme par exemple pour la restauration d'images anciennes.