Débruitage d'image par analyse en composantes principales

Livrable 3 : Rendu final

ING1 – Groupe 4

13 mai 2025

Auteurs :

EL MOUDDEN Nassef

GUERIN Enzo

GOMEZ Vincent

PHANITHAVONG Manon

RODRIGUEZ Ruben

Professeurs encadrants : FASSI Dieudonné, FORTIN Nisrine, RANISAVLJEVIC Elisabeth

Lien vers le dépôt GitHub du projet

Sommaire

1 Introduction

Dans le cadre de notre Situation d'Apprentissage et d'Évaluation (SAÉ), nous devons réaliser un projet portant sur le débruitage d'images par Analyse en Composantes Principales (ACP).

Dans un contexte pédagogique, ce projet vise à approfondir nos connaissances techniques et méthodologiques en ingénierie, tout en nous permettant de mesurer notre aptitude à organiser, mener et documenter un projet bien structuré et collaboratif. Plus précisément, l'objectif principal est d'implémenter une méthode performante de réduction de bruit dans des images numériques, en se basant sur la parcimonie, la redondance et régularité intrinsèque des données visuelles.

Cette méthode est particulièrement pertinente dans un contexte actuel où la réduction dimensionnelle est nécessaire, non seulement pour améliorer la qualité des traitements d'images, mais aussi pour répondre à des enjeux environnementaux et sociaux par la diminution du volume de données traitées.

De plus, la SAÉ présente un intérêt significatif en termes pédagogiques et professionnels puisqu'elle nous permet de développer des compétences essentielles pour notre future carrière d'ingénieur. Elle favorise notamment une approche rigoureuse de la résolution de problèmes, l'adoption de pratiques collaboratives, et l'intégration de bonnes pratiques environnementales et sociales dans les processus techniques. Notamment, le projet permettra d'évaluer spécifiquement ces cinq compétences clés :

1. Maîtriser la complexité : Abstraire un problème donné de manière scientifique.
2. Conduire un projet : adopter une démarche analytique et synthétique.
3. Collaborer et communiquer : Contextualiser et présenter le problème.
4. Innover : adopter une pensée systémique.
5. Apprendre à apprendre : Adopter une démarche réflexive.

2 Organitsation du travail

2.1 Répartition des taches

On a créé un Google Sheets pour pouvoir gérer la gestion des tâches. Cette plateforme collaborative nous permet de modifier en temps réel l'avancée du projet tout en étant sûr que toutes les tâches soient bien attribuées. De plus, on peut estimer le temps des tâches et si on voit que les délais risquent de dépasser on hésite pas à s'entraider.

QUOI ?	Livrable	QUI ?	État	QUAND ? Date de début	Date de fin
Répartition des tâches	Livrable 1	Manon	Terminé	22/04/2025	22/04/2025
Contextualisation	Livrable 1	Ruben	Terminé	22/04/2025	24/04/2025
Diagramme UML	Livrable 1	Vincent Nassef	Terminé	22/04/2025	24/04/2025
Architecture de dossiers	Livrable 1	Enzo	Terminé	22/04/2025	24/04/2025
Maquette de l'interface	Livrable 1	Manon	Terminé	23/04/2025	24/04/2025
Rapport final Livrable 1	Livrable 1	Ruben	Terminé	22/04/2025	24/04/2025
Création de l'image bruité + test et graph	Livrable 2	Enzo Manon	Terminé	22/04/2025	24/04/2025
Extraction et Vectorisation des Patchs + tes	Livrable 2	Ruben	Terminé	12/05/2025	14/05/2025
Préparation pour l'ACP + test et graph	Livrable 2	Nassef Vincent Rube	Terminé	12/05/2025	15/05/2025

FIGURE 1 – Extrait du tableur collaboratif

2.2 Outils et Librairies

— Pour la mise en commun du code nous avons utilisé **GitHub** pour sa fiabilité.
— Pour la rédaction du projet nous avons utilisé **LaTeX** pour sa qualité.
— Pour le diagramme UML nous avons utilisé **LucidApp** pour la possibilité de collaborer en temps réel.
— Pour l'interface nous avons choisis d'utiliser **Scene Builder** pour pouvoir nous rapprocher le plus fidèlement de la maquette réalisée sur **Figma** dans le *Livrable 1*.
— En terme de librairie nous avons utilisé **Apache Commons Math** pour diagonalier les matrices. **Charm-glisten-6.1.0** pour les boutons de l'interface.

3 Méthodologie

3.1 Analyse UML du projet

Le diagramme de classe fourni modélise de façon structurée l'ensemble des étapes à suivre afin de débruiter une image en utilisant des techniques de réduction de dimension, les patchs, l'ACP et le seuillage. Il expose une architecture claire et modulaire qui rend le code maintenable et évolutif.

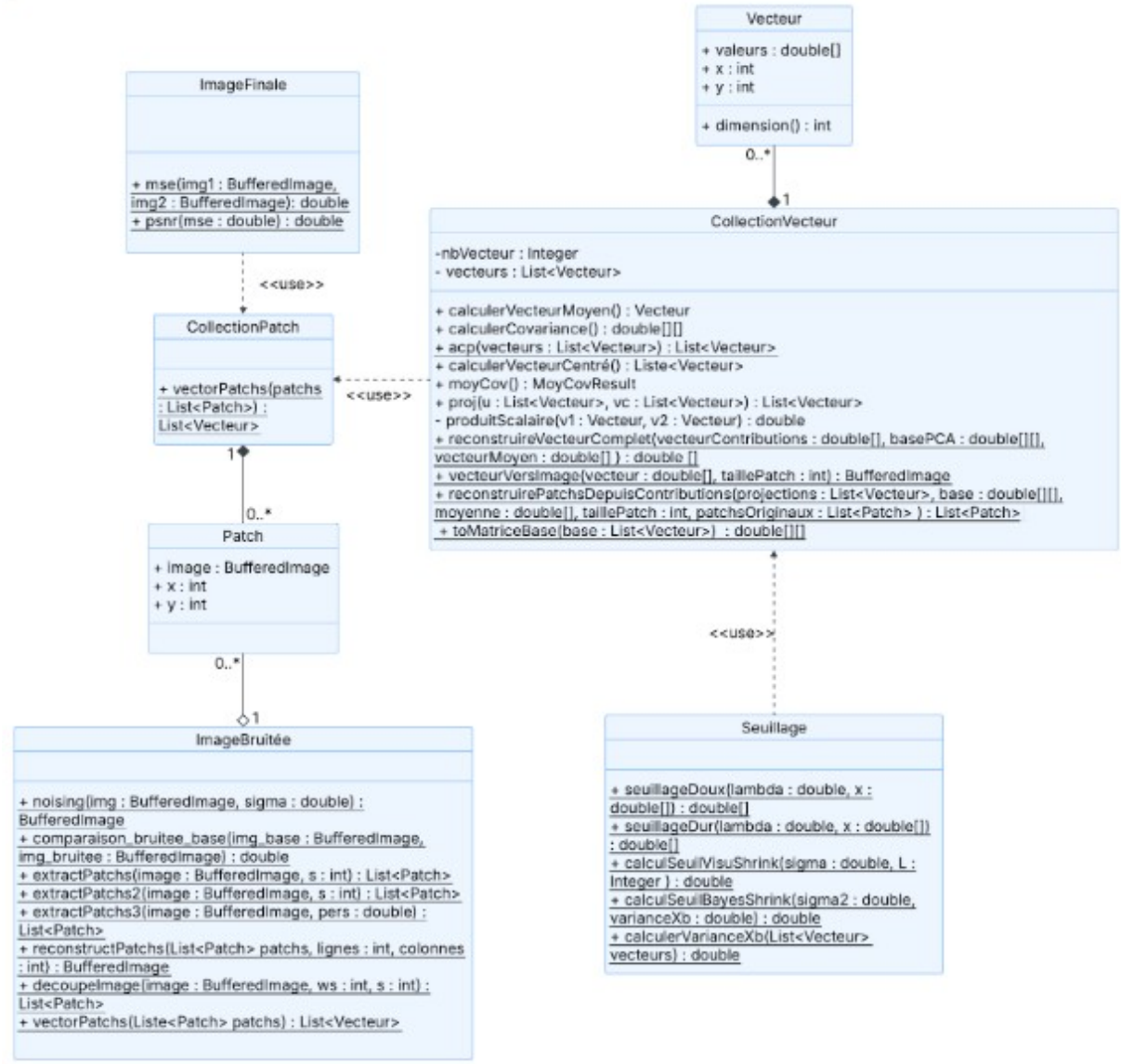

FIGURE 2 – Diagramme de classe

Par rapport aux précédents livrables, nous nous sommes rendus compte que l'on n'a pas utilisé certaines classes (Imagette Pixel, Image, Seuil). Ainsi, on a pu simplifier le diagramme mais aussi rajouter des méthodes indispensables comme par exemple produitScalaire, reconstruireVecteurComplet ou vecteurMoyen pour pouvoir les réutiliser dans les méthodes de bases.

3.2 Analyse en Composantes Principales avec seuillage

L'ACP, ou Analyse en Composantes Principales, est une méthode utilisée pour réduire la dimension d'un ensemble de données tout en gardant un maximum d'information. Mais dans le cadre de ce projet, on effectue une ACP avec seuillage. C'est-à-dire qu'on va garder les toutes les dimensions mais on va mettre certaines composantes à 0 si cces

valuers sont inférieur à ce seuil. De plus, l'ACP nous donne un avantage, elle nous donne une base orthonormée permettant de réexprimer certains vecteurs en fonction de leurs informations :

Hypothèses : $(u_1, \ldots, u_{s^2})$ forment une base orthonormée de $\mathbb{R}^{s^2}$

Montrons que :

$$V_k = m_V + \sum_{i=1}^{s^2} u_i'(V_k - m_V)u_i$$

Soit $V_k \in \mathbb{R}^{s^2}$, on part de l'identité :

$$V_k = m_V + V_k - m_V$$

Comme la famille $(u_1, \ldots, u_{s^2})$ forme une base orthonormée de $\mathbb{R}^{s^2}$, on sait que tout vecteur $x \in \mathbb{R}^{s2}$ peut s'écrire comme :

$$x = \sum_{i=1}^{s^2} \langle x, u_i \rangle u_i$$

En particulier, on applique ceci à $x = V_k - m_V$, ce qui donne :

$$V_k - m_V = \sum_{i=1}^{s^2} \langle V_k - m_V, u_i \rangle u_i$$

Ainsi :

$$\begin{aligned} V_k &= m_V + (V_k - m_V) \\ &= m_V + \sum_{i=1}^{s^2} \langle V_k - m_V, u_i \rangle u_i \\ &= m_V + \sum_{i=1}^{s^2} u_i'(V_k - m_V)u_i \end{aligned}$$

Pour les étapes du projet, on a besoin de l'ACP. Voici les étapes du projet :

— Créer une image bruitée X_b à partir d'une image d'origine X_0. Le bruit est gaussien, centré et de variance σ^2. Des cas sont à étudier si la valeur du pixel se trouve au-dessus de 255 et en-dessous de 0. On utilisera la fonction **noising**(X_0,σ).

— Extraire des patchs à partir de l'image X_b. On implémentera les fonctions **extract-Patchs**(X_b) et renvoie une collection de patchs de l'image bruitée X_b

— Vectoriser les patchs en transformant des patchs de taille s x s en un vecteur de taille s^2.

— Effectuer une Analyse en Composantes Principales à partir des vecteurs des patchs et garder les composantes gardant le maximum d'information. On utilisera une librairie qui a déjà implémenté une méthode d'ACP car si on l'implémente nous-même, on aura trop d'erreurs de calcul sur la diagonalisation de la matrice avec les

valeurs propres par rapport à un code optimisé. On cherche à avoir une image à la fin la mieux débruitée.

— Effectuer une projection dans le sous-espace des axes principaux en effectuant un seuillage. Différents seuillages et différents seuils sont à étudier comme le seuillage doux et le seuillage dur avec différents λ.

— Reconstruire une estimation de l'image X_0 à partir de la projection et seuillage.

3.3 Explication des différentes fonctions d'extraction de patches

On a implémenté 3 fonctions **extractPatchs**. On va après comparer ces méthodes sur le temps de calcul, sur le nombre de patchs extraits sur une même image et sur leur impact écologique en se supposant être une grande entreprise de la tech. Et dans un prochain livrable, on comparera ces méthodes dans le vrai cas de l'ACP pour voir si la redondance d'information est vraiment importante ou non.

3.3.1 extractPatchs1

Dans cette méthode, on extrait sur une image de taille lxc des patchs carrés de taille s*s avec s $\leq$ l et s $\leq$ c. Le premier patch est extrait à la coordonnée (0,0) et on se déplace par un décalage de 1 d'abord sur les colonnes et après avoir parcouru la première ligne, on se décale de 1 sur les lignes et on revient à la première colonne et ainsi de suite. Cette méthode représente celle avec le plus de redondance car on a ici le maximum de patches qui se croisent.

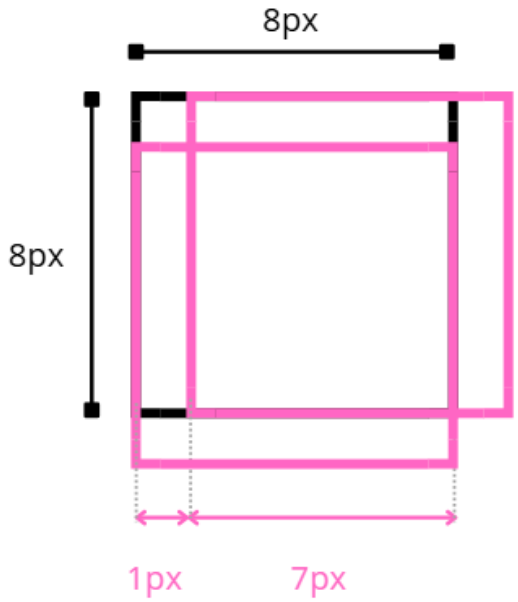

FIGURE 3 – Schéma du fonctionnement de la fonction **extrtactPatch1**

3.3.2 extractPatchs2

Dans cette méthode, on extrait sur une image de taille lxc des patchs carrés de taille s*s avec s $\leq$ l et s $\leq$ c. Le premier patch est extrait à la coordonnée (0,0) et on se déplace par un décalage de s/2 d'abord sur les colonnes et après avoir parcouru la première ligne, on se décale de s/2 sur les lignes et on revient à la première colonne et ainsi de suite. Cette méthode possède moins de redondance que la première.

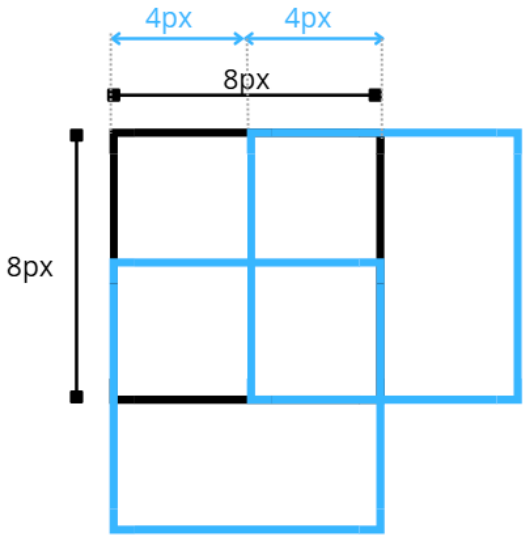

FIGURE 4 – Schéma du fonctionnement de la fonction **extrtactPatch2**

3.3.3 extractPatchs3

Dans cette méthode, on extrait sur une image de taille l x c des patchs carrés de taille cp+1 c*p+1 avec p un pourcentage passé en paramètre qui est compris entre 0 et 1. Pour la suite, on pose s=c*p+1. Le premier patch est extrait à la coordonnée (0,0) et on se déplace par un décalage de s/3 d'abord sur les colonnes et après avoir parcouru la première ligne, on se décale de s/3 sur les lignes et on revient à la première colonne et ainsi de suite. Cette méthode représente une extraction de patchs adaptatifs à l'image, les paramètres comme le pourcentage et le décalage seront étudiés dans le prochain livrable pour voir quelles valeurs sont les plus efficaces.

FIGURE 5 – Schéma du fonctionnement de la fonction **extrtactPatch3**

3.3.4 extractPatchs4

Même méthode que la **extrtactPatch2** mais ici on a une superposition de 1 pixel.

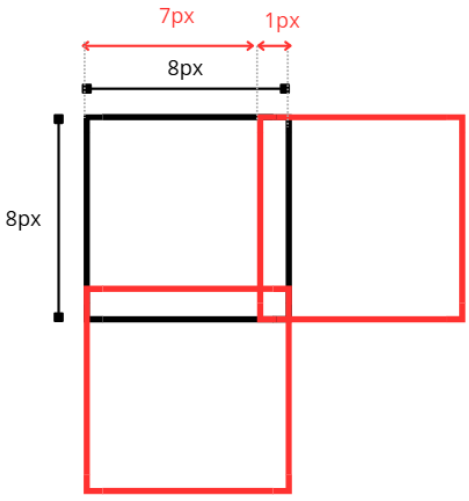

FIGURE 6 – Schéma du fonctionnement de la fonction **extrtactPatch4**

4 Tests et Résultats

Dans cette section l'objectif est de comparer les différentes méthodes d'extraction de patches puis les méthodes VisuShrink et BayesShrink en global ou local en seuillage doux

ou dur.

Dans la première partie, on s'interroge sur l'empreinte carbonne des fonctions d'extraction. Pour avoir des résultats significatifs on simule leur impact sur le long terme et une utilisation très importante.

Dans la seconde partie, on fait augmenter le bruitage de l'image (sigma) et on trace l'evolution du PSNR de l'image débruitée.

Enfin dans la troisième partie, on souhaite évaluer le rapport PSNR par temps de calcul moyenne. Grâce a cette méthode on peut trouver un compromis entre impact écologique et qualité de l'image débruitée.

4.1 Comparaison des méthodes d'extraction des patches en terme de coût environnemental

Dans cette partie, on compare les 3 méthodes d'extraction des patches. Dans chaque cas, on obtient exactement la même image que celle d'origine en reconstruisant avec les patches. Notre critère de choix se basera alors sur l'impact environnementale des différentes fonctions. Pour évaluer les performances de notre algorithme, nous avons mesuré le temps d'exécution de chaque fonction. Cette mesure a été effectuée en millisecondes (ms), en utilisant l'horloge système à l'aide de la méthode `System.currentTimeMillis()` en Java. La méthode utilisée consiste à enregistrer l'heure avant et après l'exécution de la portion de code concernée, comme suit :

```
long startTime = System.currentTimeMillis();

// Fonction à mesurer

long endTime = System.currentTimeMillis();
long executionTime = endTime - startTime;
System.out.println("Temps d'exécution : " + executionTime + " ms");
```

4.1.1 Estimation de la consommation énergétique

Pour estimer l'énergie, on utilise la formule suivante :

$$\text{Énergie (J)} = \text{durée (s)} \times \text{puissance (W)}$$

En supposant :

— un CPU consommant 25 W en charge,

— une durée de 55 ms pour `extracPatchs1` : $0.055 \times 25 = 1.375$ J,

— une durée de 20 ms pour `extracPatchs2` : $0.02 \times 25 = 0.5$ J,

— une durée de 3 ms pour `extractPatchs3` : $0.003 \times 25 = 0.075$ J,

— une durée de 4 ms pour `extractPatchs4` : $0.002 \times 25 = 0.05$ J,

4.1.2 Conversion en émission de CO_2

Hypothèse : On suppose qu'on est une grande entreprise comme Google. En 2020, on estime à 4 000 milliards de photos téléchargées. Si on suppose que l'on utilise une fonction d'extraction de patchs sur 0.1% pendant 5 ans, quel serait le coût énergétique et environnemental ? L'ADEME estime que 1 kWh équivaut à environ 450 g de CO_2 (dans le monde).

Cela donne :

$$0.1 \times 4000 \times 10^9 \times 5 = 2 \times 10^{12} \text{ exécutions}$$

Conversion en kWh des 3 fonctions :

$$\frac{1.375 \times 2 \times 10^{12}}{3.6 \times 10^6} \approx \mathbf{764000}\text{kWh}$$

$$\frac{0.5 \times 2 \times 10^{12}}{3.6 \times 10^6} \approx \mathbf{278000}\text{kWh}$$

$$\frac{0.075 \times 2 \times 10^{12}}{3.6 \times 10^6} \approx \mathbf{41700}\text{kWh}$$

$$\frac{0.05 \times 2 \times 10^{12}}{3.6 \times 10^6} \approx \mathbf{28000}\text{kWh}$$

Différence de CO_2 entre les 3 fonctions :

— Pour `extracPatches1` : $764000 \times 450 \approx \mathbf{344}$ tonnes de $\mathbf{CO_2}$,
— Pour `extracPatches2` : $278000 \times 450 \approx \mathbf{125}$ tonnes de $\mathbf{CO_2}$,
— Pour `extracPatches3` : $41700 \times 450 \approx \mathbf{18}$ tonnes de $\mathbf{CO_2}$,
— Pour `extracPatches4` : $28000 \times 450 \approx \mathbf{12,5}$ tonnes de $\mathbf{CO_2}$.

4.1.3 Conclusion

Une optimisation algorithmique peut paraître négligeable à petite échelle, mais à l'échelle d'une infrastructure distribuée ou d'un service utilisé massivement, les économies en énergie et en CO_2 peuvent être considérables. C'est pourquoi on souhaite conserver la fonction `extracPatches3`.

4.2 Comparaison des différents seuillages

Graphique représentant le MSE en fonction du niveau de seuillage

4.3 Comparaison des méthodes d'extraction et des méthodes Local, Global après ACP avec des seuillages doux et dur

Dans cette partie, nous allons comparer les méthodes d'extraction après l'ACP. Pour cela, on utilise l'erreur quadratique moyenne (MSE) et le rapport signal sur bruit (PSNR). Comme les deux sont liés (inversement proportionnel) on utilisera que le PSNR que l'on devra maximiser. Pour voir comment l'évolution du PSNR en fonction du bruit se comporte, on a choisi d'afficher les graphiques en échelle logarithmique. Ainsi, pour préciser, les valeurs pour VisuShrink sont comprises entre 20 et 30.

4.3.1 extractPatchs1

D'après le calcul du PSNR, on remarque que la pire méthode est en Local doux et la meilleure en Global Dur pour la méthode `extractPatchs1`.

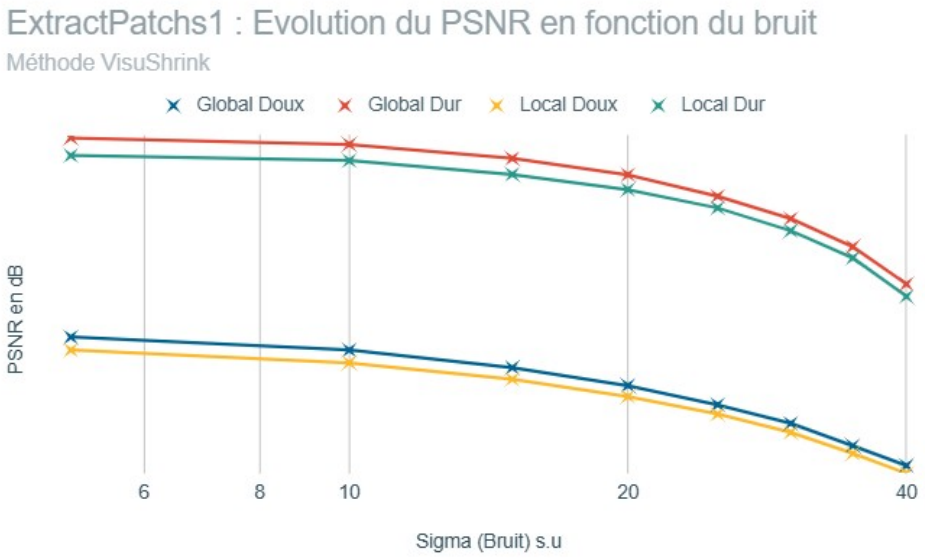

FIGURE 7 – Graphique en courbe représentant le PSNR des méthodes avec VisuShrink

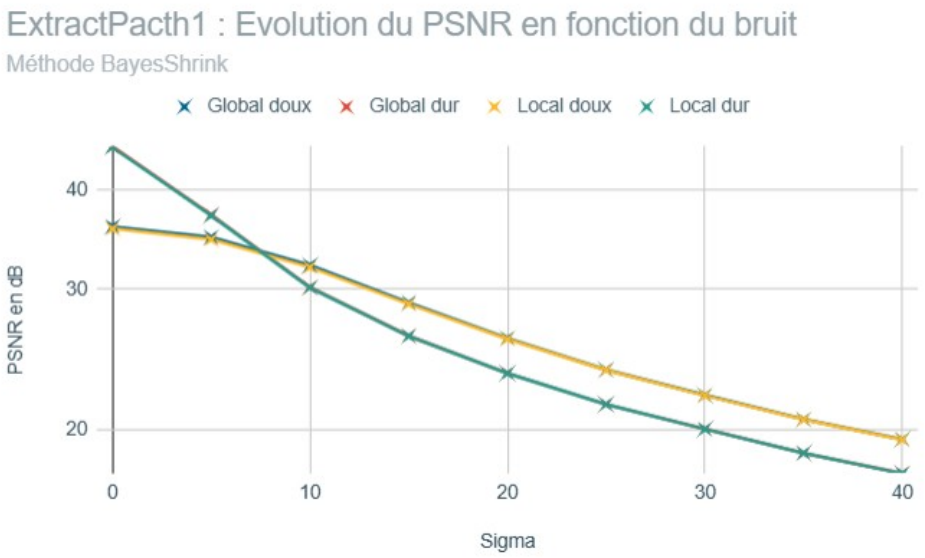

FIGURE 8 – Graphique en courbe représentant le PSNR des méthodes avec BayesShrink

FIGURE 9 – Image débruitée au pire (Local Doux)

FIGURE 10 – Image débruitée au mieux (Global Dur)

4.3.2 extractPatchs2

D'après le calcul du PSNR, on remarque que la pire méthode est en Global Doux et la meilleure en Local Dur pour la méthode `extractPatchs2`.

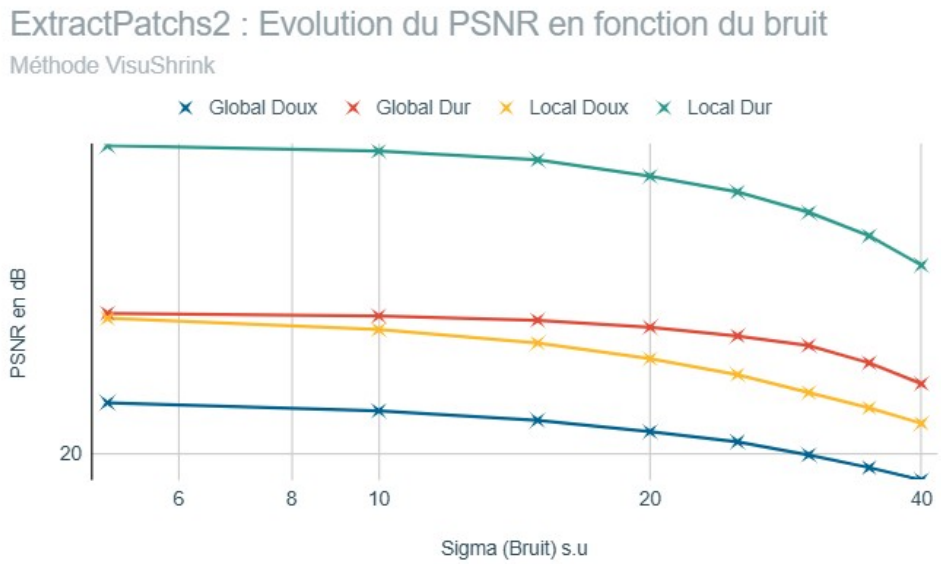

FIGURE 11 – Graphique en courbes représentant le PSNR des méthodes avec VisuShrink

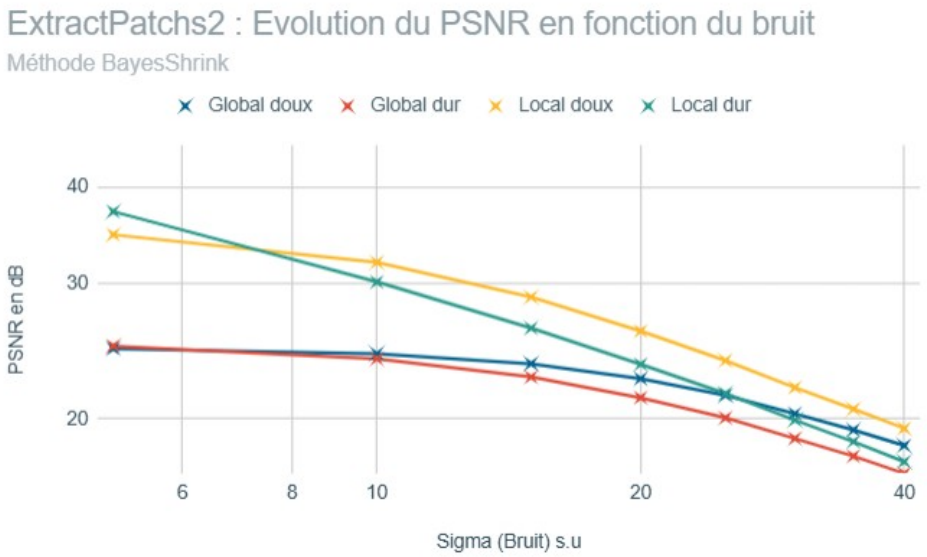

FIGURE 12 – Graphique en courbes représentant le PSNR des méthodes avec BayesShrink

FIGURE 13 – Image débruitée au pire (Global Doux)

FIGURE 14 – Image débruitée au mieux (Local Dur)

4.3.3 extractPatchs3

D'après le calcul du PSNR, on remarque que la pire méthode est en Global Doux et la meilleure en Local Dur pour la méthode `extractPatchs3`. C'est normal d'obtenir ces résultats, car `extractPatchs3` n'est qu'une version simplifiée de `extractPatchs2`.

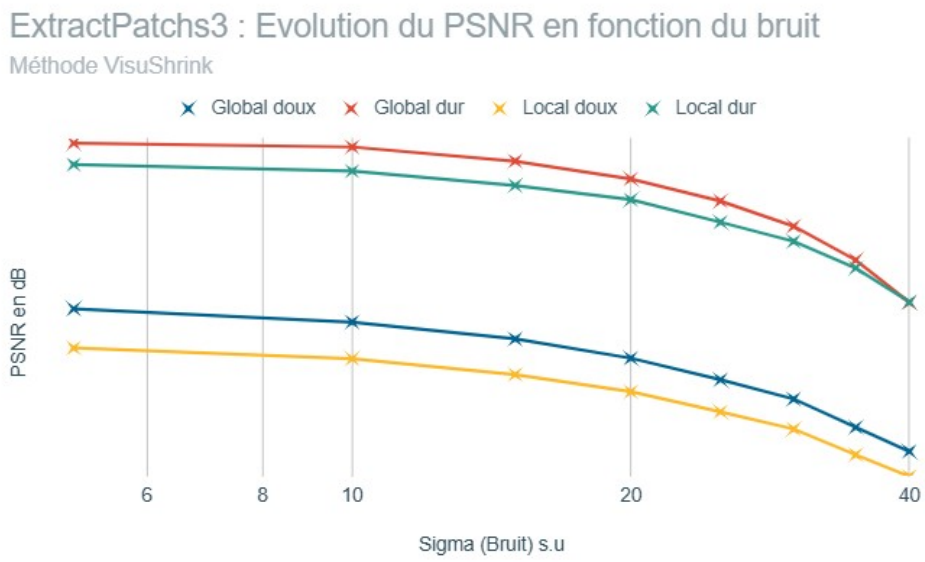

FIGURE 15 – Graphique en courbes représentant le PSNR des méthodes avec VisuShrink

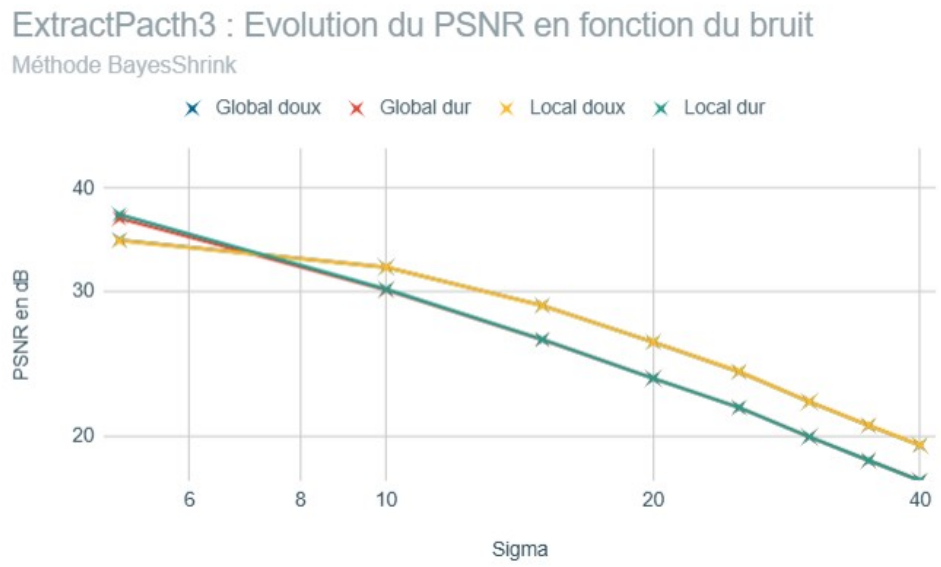

FIGURE 16 – Graphique en courbes représentant le PSNR des méthodes avec BayesShrink

FIGURE 17 – Image débruitée au pire (Global Doux)

FIGURE 18 – Image débruitée au mieux (Local Dur)

4.3.4 extractPatchs4

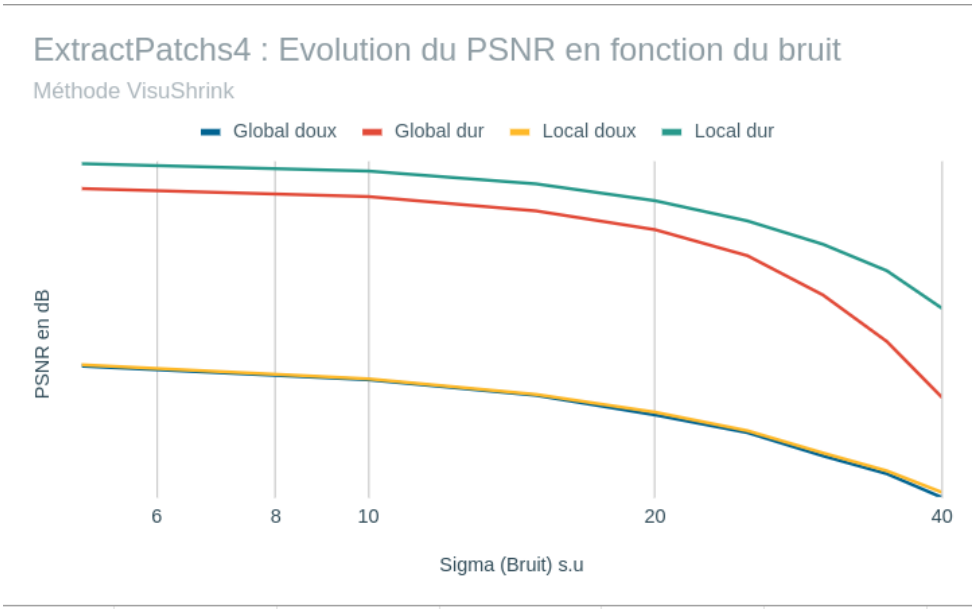

FIGURE 19 – Graphique en courbes représentant le PSNR des méthodes avec VisuShrink

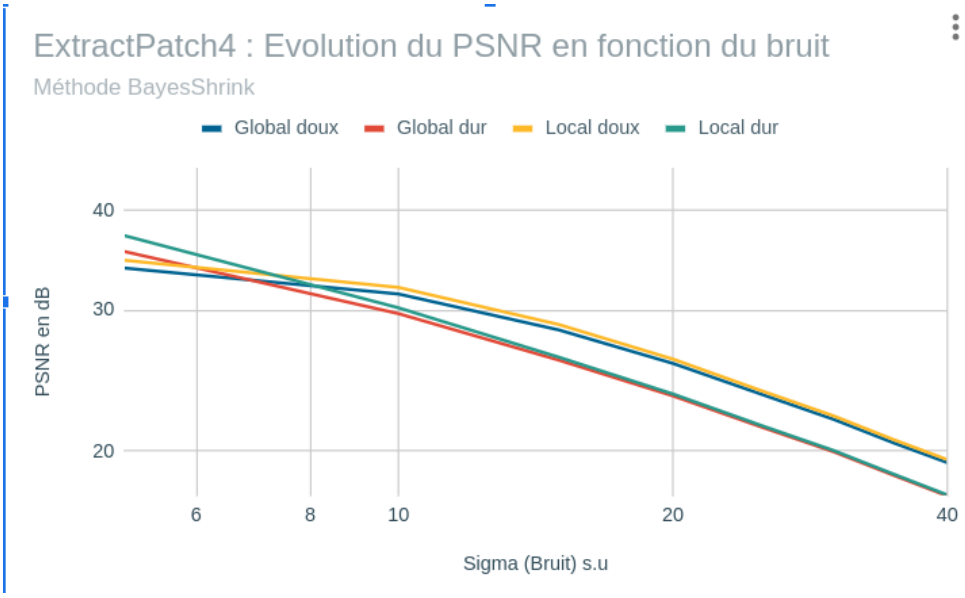

FIGURE 20 – Graphique en courbes représentant le PSNR des méthodes avec BayesShrink

FIGURE 21 – Image débruitée au pire (Global Doux)

FIGURE 22 – Image débruitée au mieux (Local Dur)

4.3.5 Conclusion

On peut déduire plusieurs choses des graphiques obtenus pour les différentes méthodes d'extractPatchs. Premièrement, on obtient que les 2 méthodes de seuillages fonctionnent mieux sur des intervalles de sigma différents : BayesShrink fonctionne mieux sur des seuillages avec $\boldsymbol{\sigma < 20}$ et VisuShrink pour des $\boldsymbol{\sigma > 20}$ que ce soit pour tout les extractPatchs différents.

Aussi, on obtient que le facteur de redondance augmente très légèrement le $\boldsymbol{PSNR}$, ceci augmente d'environ d'un maximun de 0.2 à 0.5 en général à sigma fixé entre les méthode à part pour extractPatch1 qui obtient un PSNR très grand par rapport aux autres pour $\boldsymbol{\sigma = 0}$, ce qui ne compte pas vraiment car dans ce cas-là, il n'y aurait pas de bruitage sur l'image.

Enfin, moins il y a de redondance et plus les méthodes locales sont efficaces, ceci est logique car dans les méthodes d'ACP locales, on cherche à ressortir le plus d'information sur chaque patch alors que s'il y a de la redondance, on doit faire une moyenne des composantes superposées et on pert l'aspect local.

4.4 Comparaison du temps d'exécution et de la qualité de reconstruction

Dans cette partie, nous comparons les différentes méthodes d'extraction de patchs en fonction du **PSNR** obtenu pour un bruit de variance fixée à $\boldsymbol{\sigma = 25}$. Cette valeur a été choisie car elle représente un niveau de bruit suffisamment élevé pour rendre le débruitage pertinent, tout en permettant de se concentrer sur un seul cas de test représentatif.

Le seuillage est déterminé à l'aide de la méthode **VisuShrink**, qui s'avère particulièrement efficace pour les niveaux de bruit élevés.

4.4.1 Tableau de comparaison

Méthode	Temps extract (ms)	Global doux (ms)	Global dur (ms)	Découpe local	Local doux (ms)	Local dur (ms)
`extractPatch1`	55	201	211	295	128	122
`extractPatch2`	20	18	15	309	199	190
`extractPatch3`	3	13	12	1088	446	411
`extractPatch4`	4	10	7	321	149	131

TABLE 1 – Comparaison des temps d'extraction et de traitement selon les méthodes d'extraction de patchs (taille = 8, $\sigma = 25$)

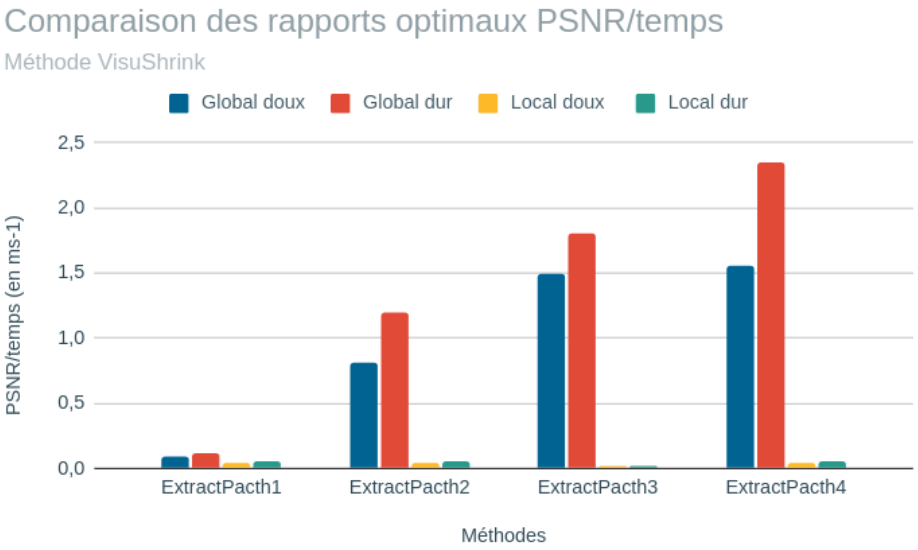

FIGURE 23 – Graphique en bâtons représentant le rapport entre le PSNR et le temps de calcul pour les différentes méthodes

On remarque qu'à partir de la taille de patchs 8 x 8, le PSNR commence à baisser.

On remarque qu'ici, le temps pour effectuer l'ACP global est plus court pour les méthodes d'extraction des patchs.

Ainsi, on a calculer le rapport entre le PSNR et le temps moyen pour voir quelles méthodes étaient les plus efficaces.

4.4.2 Conclusion

On remarque avec ce graphique que le petit gain de PSNR apporté par la redondance est négligeable par rapport au temps gagné à l'éxécution et que même si la méthode local dur s'avère être la plus efficace en gain de PSNR, elle est des dizaines de fois plus longue que la méthode global dure. On gardera dans ce cas-là la méthode d'extractPatchs4 pour notre interface.

4.5 Comparaison du psnr en fonction de la taille des patchs

D'après nos différents résultats, on obtient que la méthode extractPatch4 est plus efficace en moyenne que les autres méthodes sur toutes les différentes méthodes d'ACP. Il faut maintenant comparer quelle taille de patch est la plus efficace.

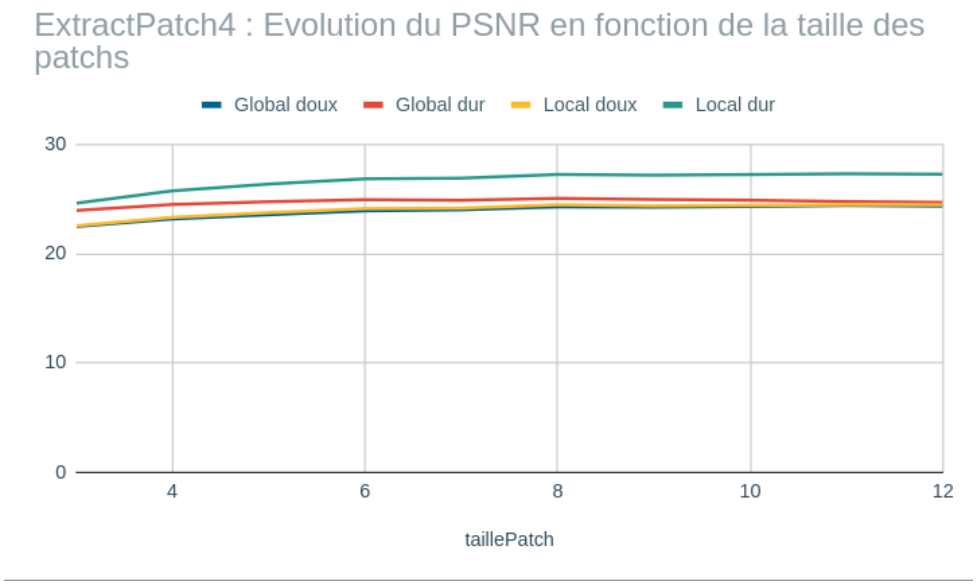

FIGURE 24 – Graphique en courbes représentant l'évolution du PSNR en fonction de la taille des Patchs

On remarque qu'à partir de la taille de patchs 8 x 8, le PSNR commence à baisser pour la méthode d'ACP globale dure. On choisit donc de garder la taille des patchs à 8 parce que les utilisateurs auront le droit de choisir leurs méthodes et donc aucune n'est privilégiée.

5 Interface

D'après le `Livrable 1` nous avions décidé de l'apparence globale de l'IHM. Un menu violet fonctionnel et interactif,il récapitule les étapes en cours. Nous avons pensé à une option de zoom afin de voir en détail notre image mais aussi anticiper la possibilité de mettre une image de n'importe quelle taille. Nous avons décider d'enlever le bouton collection pensant qu'il n'avait pas tant d'utilité que cela.

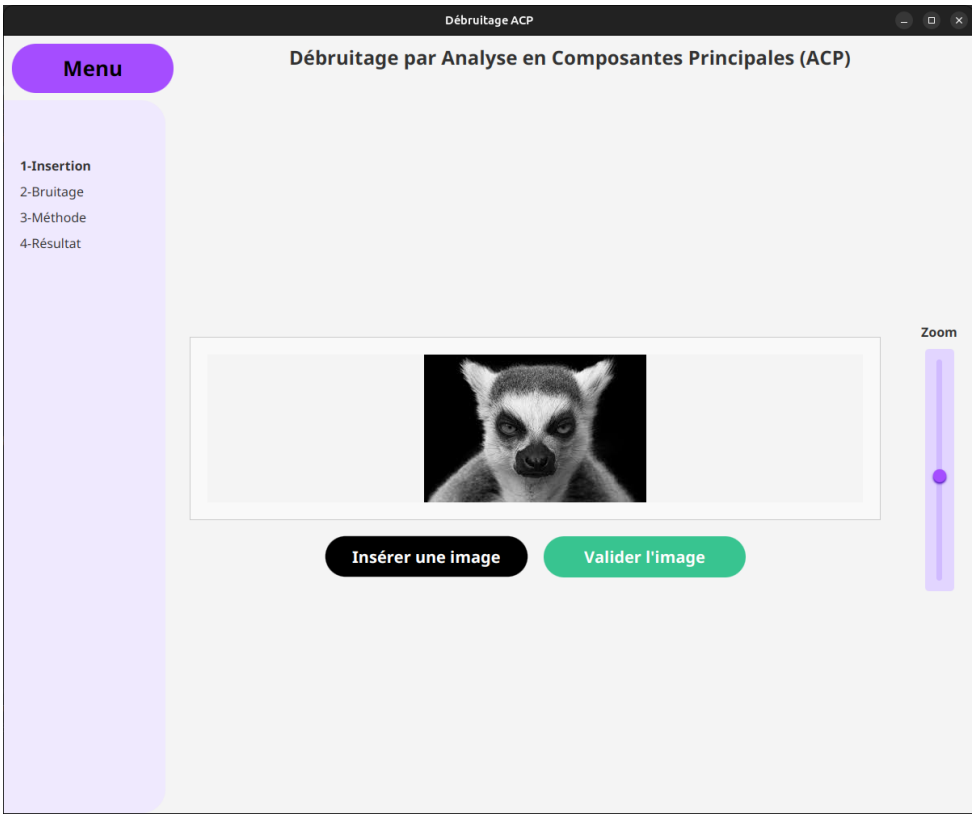

FIGURE 25 – Page d'insertion de l'image par l'utilisateur

L'utilisateur arrive sur cette page, il ne peut faire qu'une seule chose insérer une image depuis son ordinateur. Ensuite, il peut zoomer ou dézoomer son image en fonction si on voit pas assez les détails et il peut valider l'image pour aller à l'étape de bruitage de l'image.

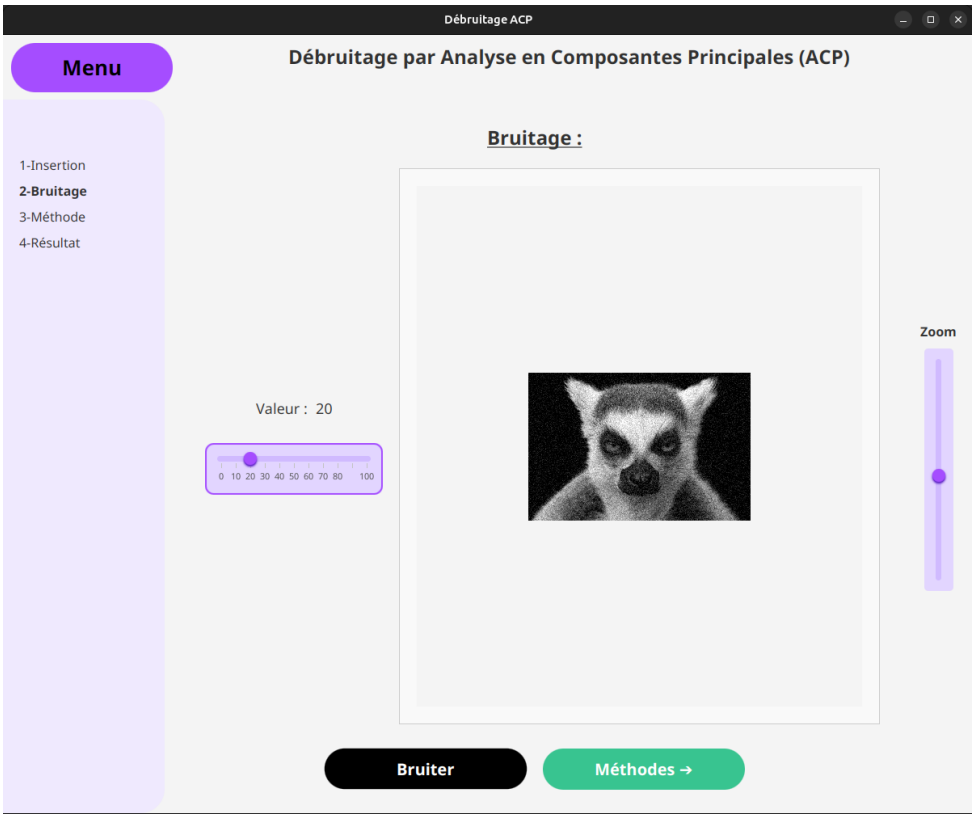

FIGURE 26 – Page de bruitage

Sur cette page, on a toujours l'image affichée avec l'option de zoom. On a placé un slider qui permet de calibrer le degré de bruitage voulu. Le bouton bruiter permet d'appliquer le bruitage à l'image et d'afficher l'image bruitée. Ensuite l'utilisateur peut appuyer appuyer sur le bouton "méthodes" qui permet de nous diriger sur la page des différentes méthodes de débruitage.

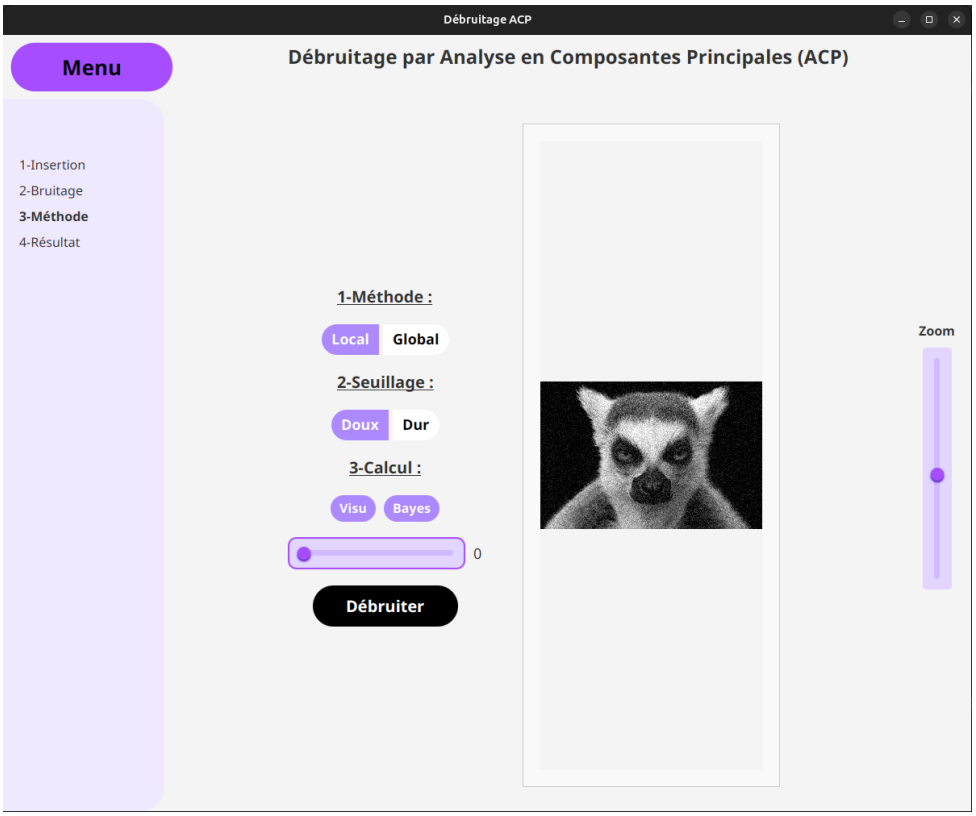

FIGURE 27 – Page de débruitage

Sur cette page, l'utilisateur peut choisir la méthode d'ACP avec le choix soit local, soit global. Ensuite, il peut choisir le seuillage et à la fin, il peut choisir le seuil soit avec le slider, soit avec les boutons Visu ou Bayes qui calculent le seuil appropré en fonctions de la méthode choisie et modifie la valeur du slider. S'il a fini de choisir, il peut appuyer sur le bouton "débruiter".

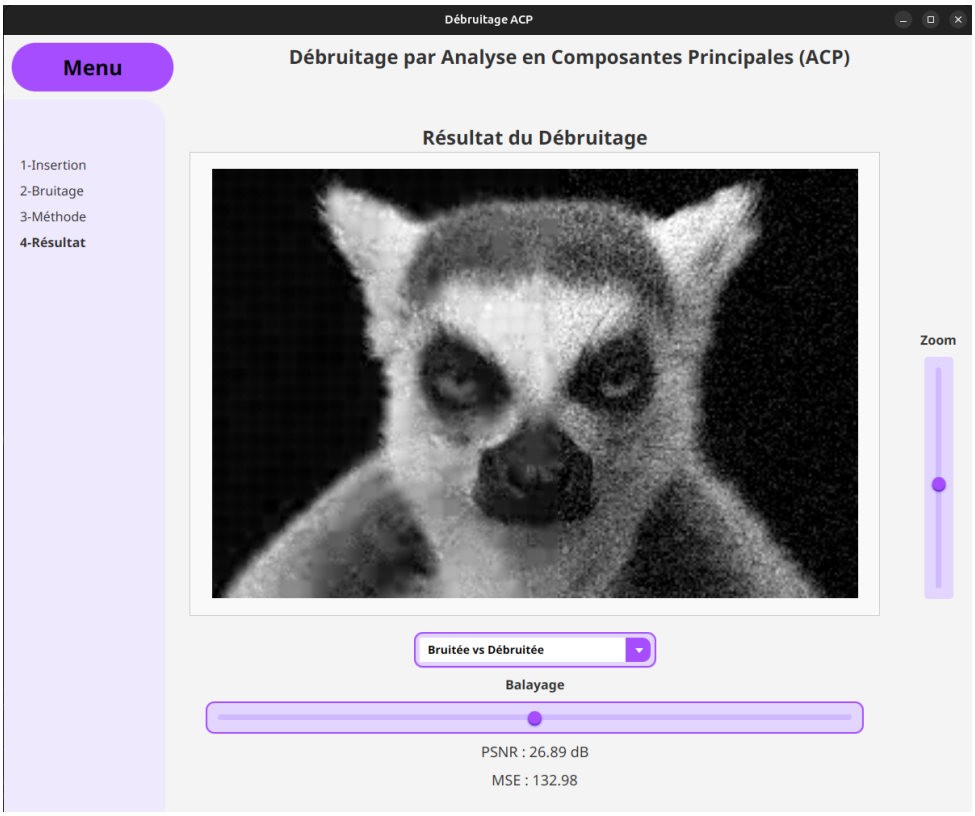

FIGURE 28 – Page de résultats

Enfin, l'utilisateur arrive sur la page de résultat. Il y a un slider qui permet de défiler les pixels de la gauche vers la droite entre l'image bruitée et l'image débruitée. Cette option permet à l'utilisateur d'être surpris du résultat obtenu en découvrant petit à petit l'image obtenue. On a aussi la valeur du PSNR de l'image débruitée par rapport à l'image d'origine.

6 Pour aller plus loin

Une perspective intéressante pour ce projet serait d'étendre la méthode de débruitage par ACP aux images en couleur. Cela impliquerait de traiter séparément ou conjointement les canaux **RGB** tout en adaptant les algorithmes pour préserver les informations chromatiques. L'impact environnemental devra être réévalué, car le traitement des images en couleur nécessitera davantage de calculs, augmentant potentiellement la consommation énergétique et les émissions de CO.

De plus, pour mettre en avant l'impact énergétique et écologique, il aurait été préférable d'utiliser un outil comme **Code Carbone** très pratique à intégrer directement dans le code, il permet de voir la consommation réel en fonction de notre position géographique.

7 Retour sur le travail d'équipe

Ce projet a été une expérience enrichissante sur le plan collaboratif. La répartition des tâches, organisée via un tableur partagé, a permis une bonne visibilité sur l'avancement de chacun et une réactivité en cas de besoin. Nous avions réparti les tâches avant chaque livrable en mettant en avant les compétences de chacun.

La diversité des compétences au sein du groupe a été un atout : certains membres étaient plus à l'aise avec les aspects mathématiques (comme l'ACP), tandis que d'autres excellaient en programmation ou en conception d'interface. De plus, on a constaté que faire l'effort pour le *Livrable 1* de faire une maquette nous a beaucoup aidés à visualiser en amont le résultat voulu.

Les défis principaux ont concerné la gestion des informations et la cohérence des implémentations (par exemple, les différentes méthodes d'extraction de patches).En effet, le nombre de fonctions différentes bien qu'intéressantes nous a compliqué la partie des tests. Cependant, la phase de tests et d'analyse des résultats a renforcé notre esprit critique collectif. Enfin, on pense avoir réussi à mettre en avant l'efficacité des méthodes tout en considérant leur impact environnemental.

En résumé, ce travail a souligné l'importance de la communication et de la flexibilité dans un projet collaboratif. Si c'était à refaire, nous accorderions plus de temps aux tests intermédiaires et à la documentation partagée pour fluidifier encore le processus.

8 Bibliographie

— Wikipedia contributors, *Google Photos*, Wikipedia. Disponible à : https://en.wikipedia.org/wiki/Google_Photos.

— *Définition de l'ACP*, Appviser. Disponible à : https://www.appvizer.fr/magazine/technologie/editeurs/acp-glossaire.

— Groupe 4, *Livrable 1 : Analyse du projet*. Disponible à : https://docs.google.com/document/d/1n1c1Wc7xWqYphTxk9MSXAvmsZzUdiJ5Cpoob_ioLVeE/edit?usp=sharing

— ScienceDirect, *Peak Signal-to-Noise Ratio*. Disponible à : https://www.sciencedirect.com/topics/computer-science/peak-signal-to-noise-ratio.

— CodeCarbone team, *Code Carbone*. Disponible à : https://codecarbon.io/.