Guide de Test et Documentation des Travaux Pratiques
Traitement d'Images

Donfack Pascal
Classe : M2-GI

22 décembre 2025

1 Introduction

Ce document constitue le guide de test exhaustif pour les travaux pratiques (TP) de traitement d'images. Chaque algorithme a été implémenté avec une rigueur académique, privilégiant la manipulation **manuelle** des pixels (via des boucles imbriquées) au détriment des fonctions pré-établies des bibliothèques de haut niveau. Cette approche garantit une compréhension profonde des transformations matricielles appliquées aux images.

Un aspect crucial de cette révision est l'intégration systématique de la **visualisation directe**. Chaque script génère désormais un canevas interactif (via Matplotlib) permettant de comparer instantanément l'image source et le résultat du traitement.

2 Architecture du Projet

La structure du projet respecte une organisation modulaire par type de traitement :
- **inputs/** : Répertoire central des images de test (`img1.png` et `img2.png`).
- **Chap1/ à Chap7/** : Dossiers thématiques regroupant les scripts Python (`.py`).
- **outputs/** : Sous-répertoires au sein de chaque chapitre contenant les fichiers sauvegardés après exécution.

3 Protocole de Test

Pour valider le fonctionnement d'un TP, l'utilisateur doit disposer d'un environnement Python configuré avec les bibliothèques suivantes : `numpy`, `pillow`, `matplotlib`.

3.1 Exécution

La commande type pour lancer un TP est la suivante :

```
python ChapX/tpX_nom_du_fichier.py
```

À l'exécution, une interface graphique s'affiche. L'utilisateur peut ainsi observer :
- L'image originale (souvent convertie en niveaux de gris pour le traitement).
- Le résultat de l'algorithme (filtrage, segmentation, transformée, etc.).

— Des aides visuelles (histogrammes de fréquence, spectres de magnitude, zones de croissance, etc.).

4 Détail des Chapitres

4.1 Chapitre 1 - Fondamentaux

Ce chapitre couvre les bases : conversion en niveaux de gris, échantillonnage spatial, quantification et profils d'intensité.

— `tp1_rgb_to_gray.py` : $Conversion pondre manuelle RGB vers Niveaux de Gris$.
- `tp1_gray_quantization.py` : $Rduction du nombre de niveaux d'intensit (quantification sur K bits)$.

- `tp1_spatial_sampling.py` : $Rduction de la rsolution spatiale par sous - chantillonnage$.
- `tp1_intensity_profile.py` : $Extraction et trac de la ligne d'intensit au milieu de l'image$.

4.2 Chapitre 2 - Traitements Ponctuels

Calcul d'histogrammes, étirement de contraste, correction gamma et égalisation. Les scripts affichent systématiquement l'histogramme avant/après.

— `tp2_histogram.py` : $Calcul et affichage de l'histogramme de frquences$.
- `tp2_stats.py` : $Calcul des statistiques descriptives (Moyenne, cart-type, Min, Max)$.

- `tp2_linear_stretch.py` : $tirement linaire du contraste de [min, max] vers [0, 255]$.
- `tp2_linear_saturation.py` : $tirement avec seuils de saturation pour liminer les valeurs aberrantes$.

- `tp2_piecewise_linear.py` : $Transformation par morceaux pour un contrle fin sur des plages d'intensit$

- `tp2_gamma_correction.py` : $Correction de gamma via une table de correspondance (LUT)$.

- `tp2_hist_equal.py` : $galisation globale base sur la fonction de rpartition (CDF)$.
- `tp2_local_hist_equal.py` : $galisation adaptative locale (fentre 15x15)$.
- `tp2_arithmetic_ops.py` : $Oprations de fusion, addition et soustraction d'images$.
- `tp2_logical_ops.py` : $Oprations boolennes (AND, OR) sur images binaires$.
- `tp2_interp_nearest.py` : $Zoom par plus proche voisin$.
- `tp2_interp_bilinear.py` : $Zoom par interpolation bilinaire (plus lisse)$.
- `tp2_interp_bicubic.py` : $Zoom par interpolation bicubique (spline de Catmull - Rom)$.

4.3 Chapitre 3 - Filtrage de Convolution

Implémentation d'un moteur de convolution 2D manuel.

— `tp3_convolution_2d.py` : $Moteur de convolution gnral (utilis pour le Sharpening)$.
- `tp3_mean_filter.py` : $Filtrage moyenneur pour le lissage uniforme$.
- `tp3_gaussian_filter.py` : $Flou gaussien (lissage prservant mieux les structures)$.
- `tp3_median_filter.py` : $Filtrage statistique pour supprimer le bruit impulsionnel (Salt\&Pepper)$.

4.4 Chapitre 4 - Domaine Fréquentiel

Transformée de Fourier Discrète (DFT). Pour des raisons de performance, la DFT 2D est calculée sur un patch de 64x64.

— `tp4_dft_1d.py` : $Calcul de la transformee de Fourier sur un signal sonore/temporel.$
- `tp4_dft_2d.py` : $Analyse spectrale sur un patch d'image (log - magnitude).$
- `tp4_fft_shift.py` : $Centrage des basses frquences au centre du spectre.$
- `tp4_lowpass_freq.py` : $Lissage par suppression des hautes frquences dans le domaine spectral.$

- `tp4_highpass_freq.py` : $Extraction de dtails par suppression des basses frquences.$
- `tp4_notch_filter.py` : $Rejet de bande pour liminer des bruits priodiques cibls.$
- `tp4_enhance_highpass.py` : $Accentuation des dtails (High - boost filtering).$

4.5 Chapitre 5 - Détection de Contours

Opérateurs Sobel, Prewitt, et Laplacien.

— `tp5_gradient_finite_diff.py` : $Calcul des drives premires simples (diffrences finies).$
- `tp5_sobel.py` : $Dtection de contours par oprateurs de Sobel (plus robuste).$
- `tp5_prewitt.py` : $Variante de Sobel utilisant des poids uniformes.$
- `tp5_roberts.py` : $Oprateur de gradient croi (2x2).$
- `tp5_laplacian.py` : $Dtection par drives secondes (Laplacien).$
- `tp5_edge_threshold.py` : $Binarisation des contours par seuillage du gradient.$
- `tp5_non_max_suppression.py` : $Affinement des contours par suppression des non-maxima.$

- `tp5_hysteresis.py` : $Liaison des contours faibles par double seuillage (Canny - style).$

4.6 Chapitre 6 - Segmentation

Seuillage d'Otsu, croissance de régions et K-means.

— `tp6_simple_threshold.py` : $Binarisation globale seuil fixe.$
- `tp6_otsu_threshold.py` : $Calcul automatique du seuil optimal par maximisation de la variance inter-classe.$
- `tp6_region_growing.py` : $Segmentation par agrgation de pixels depuis un germe (seed).$
- `tp6_connected_components_labeling.py` : $Identification et coloration des objets distincts.$

- `tp6_kmeans_segmentation.py` : $Segmentation par clustering de couleurs (K-means).$

- `tp6_watershed.py` : $Segmentation par ligne de partage des eaux sur le gradient.$

4.7 Chapitre 7 - Morphologie Mathématique

Opérations sur images binaires (Érosion, Dilatation, Ouverture, Fermeture).

— `tp7_connected_components_labeling.py` : $tiquetage binaire pour l'analyse d'objets.$
- `tp7_freeman_chain_code.py` : $Encodage compact des contours d'un objet.$
- `tp7_morph_erosion.py` : $rosion (rduction des formes, suppression du bruit fin).$
- `tp7_morph_dilation.py` : $Dilatation (expansion des formes, comblement de trous).$
- `tp7_morph_opening.py` : $rosion suivie d'une dilatation (nettoyage).$
- `tp7_morph_closing.py` : $Dilatation suivie d'une rosion (fermeture de cavits).$

- `tp7`$_{morph_gradient.py}$: $Extraction des contours par di ff rence entre dilatation et rosion.$

5 Conclusion

Tous les scripts sont conçus pour être simples à lire et à tester. La visualisation directe permet de valider instantanément la correction de la logique implémentée. Ce projet démontre l'efficacité des méthodes fondamentales de traitement d'images sans recours à des bibliothèques externes complexes.

Annexe : Répertoire du Projet

L'intégralité du code source, des images de test et des résultats est disponible sur GitHub à l'adresse suivante :
`https://github.com/Tiger-Foxx/images-treatment-practice.git`