

Master 1 Informatique

UE-INF1207M : Informatique Graphique et Image

Partie - Analyse d'image

TP 1 — Implémentation des traitements fondamentaux

Préambule

L'objectif de ce premier TP est de comprendre les bases du traitement d'image numérique par l'implémentation directe des algorithmes fondamentaux.

Vous êtes explicitement invités à écrire du code manipulant les pixels, les histogrammes et les transformations de base. L'utilisation de fonctions toutes faites est volontairement limitée afin de favoriser une compréhension approfondie des mécanismes sous-jacents.

Installation d'OpenCV

Vous devez installer OpenCV sur votre machine avant la séance de TP.

Version recommandée : OpenCV 4.12.0 (année 2025).

Cependant, toute version valide de la branche **4.x** est acceptée.

Pour ce TP, OpenCV doit être utilisé **exclusivement** pour la lecture et l'écriture d'images, l'affichage, et la structure de données `cv::Mat`.

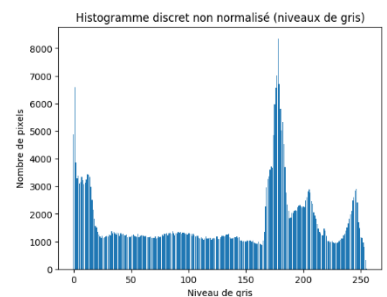
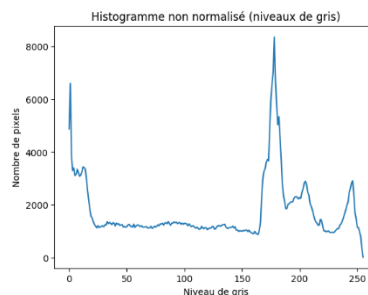
Règles d'utilisation d'OpenCV

Fonctions autorisées : `imread`, `imwrite`, `imshow`, `waitKey`, `cv::Mat` et accès mémoire bas niveau.

Toute fonction OpenCV réalisant un traitement d'image est strictement interdite pour ce TP et entraînera un malus !!!

5 — Transformations d'histogramme

l'histogramme nous informe sur la répartition des niveaux de gris (dynamique, contraste, etc.), ainsi que sur les niveaux de gris les plus représentés (modes). Il est cependant dépourvu de toute information spatiale. généralement, on normalise ses valeurs par le nombre total de pixels de l'image. Dans ce cas, l'histogramme peut être vu comme une distribution de probabilités.



Nous avons vu en cours les différentes transformations d'histogramme permettant de modifier l'apparence d'une image en niveaux de gris (voir le support de cours fourni).

Dans cette section, vous devez implémenter manuellement ces transformations et analyser leur effet sur l'image et sur l'histogramme associé.

Pour chacune des transformations demandées, vous devrez afficher l'image résultante, afficher l'histogramme associé, et commenter qualitativement les résultats.

5.1 — Inversion de l'histogramme

Implémenter une transformation visant à inverser la répartition des niveaux de gris de l'image.

- Analyser l'effet de cette inversion sur l'image et sur la symétrie de l'histogramme.
- Comparer visuellement cette transformation avec l'image originale.

5.2 — Expansion, compression et translation de l'histogramme

Implémenter des transformations permettant de modifier l'étendue et la position de l'histogramme.

- Vous devrez proposer/implémenter une transformation d'expansion,
- une de compression et une de translation.
- Pour chaque cas, analyser l'impact sur le contraste, la luminosité globale et la forme de l'histogramme (affichage de l'image résultat)

5.3 — Égalisation d'histogramme

Implémenter une transformation visant à redistribuer les niveaux de gris afin d'obtenir une distribution plus uniforme.

- Comparer l'image et l'histogramme avant et après égalisation.
- Discuter dans quels cas l'égalisation améliore l'image et dans quels cas elle peut la dégrader.

6 — Transformations par convolution

Contrairement aux transformations ponctuelles, les transformations par convolution exploitent le voisinage spatial de chaque pixel.

Nous avons vu en cours le principe général de la convolution discrète appliquée aux images.

Dans cette section, vous devez implémenter la méthode générale de convolution puis l'appliquer à différents cas d'étude.

6.1 — Implémentation de la convolution discrète (méthode générale)

Implémenter une méthode générique de convolution appliquée à une image en niveaux de gris.

Votre implémentation devra gérer explicitement le voisinage des pixels et le traitement des bords.

Discuter l'impact du choix de la gestion des bords sur le résultat.

6.2 — Application : lissage de l'image

À l'aide de la méthode de convolution implémentée précédemment, appliquer un noyau de lissage.

Analyser l'effet du lissage sur les détails fins, le bruit et la distribution des niveaux de gris.

Comparer l'histogramme avant et après lissage.

6.3 — Application : rehaussement des contours

À l'aide de la méthode de convolution implémentée précédemment, appliquer un noyau permettant de mettre en évidence les contours.

Analyser l'effet sur les transitions d'intensité et sur l'histogramme.

Comparer ces résultats avec ceux obtenus par les transformations ponctuelles.

Question de synthèse bonus

Comparer les transformations ponctuelles et les transformations par convolution en termes d'information exploitée, d'effet visuel et d'impact sur l'histogramme.

Expliquer pourquoi deux images peuvent présenter des histogrammes similaires tout en étant visuellement très différentes.

Livrables attendus

- Code C++ commenté (avec structures de données explicités !)
- Images générées (résultats et histogrammes)
- Rapport d'analyse (3 à 4 pages maximum).
- Une analyse de la complexité sera appréciée (et traduite dans la note).

Note importante

- Une démonstration du TP sera faite en présence de l'un des encadrant du TP. La présence des deux membres du binôme est requise (compter 10mn pour la démo/binôme). L'absence à la séance de démo est considérée comme une absence à un examen.
- Le but de cet oral est de s'assurer que les membres du binôme ont participé à la réalisation du TP, et que la production est bien personnelle !

- ANNEXE :

Contraintes fortes imposées aux étudiants

- Langage **C++ uniquement**
- Utilisation **exclusivement** d'OpenCV 4.x (à partir de version 4.8 en 2025-2026)
- **CMake** obligatoire pour la compilation (fichier CMakeLists.txt à fournir)

Indication minimaliste : pour OpenCV, vous devez ajouter les #include suivants :

```
#include <opencv2/core.hpp>
#include <opencv2/imgcodecs.hpp>
#include <opencv2/highgui.hpp>

#include <iostream>
#include <vector>
#include <cmath>
#include <algorithm>
#include <string>
```

Toute bibliothèque incluse dans votre code doit être justifiée par son utilisation effective.

Petit test pour votre installation :

```
int main() {
    cv::Mat img = cv::imread("test.jpg");
    if (img.empty()) return 1;
    cv::imshow("test", img);
    cv::waitKey(0);
    return 0;
}
```