Université Clermont Auvergne

École Universitaire de Physique et d'Ingénierie

Diplôme : Master Imagerie et Technologie pour la Médecine (TechMed)



Travaux Encadré de Recherche

**Présenté par :** LEHARA Lyes

**Thème :** Bibliothèque de traitement d'images en C+

**Dirigé par :** Mme. PERY Emile

**Année universitaire :** 2024/2025.

Table de matière

Table des matières

[**1.** **Introduction :** 4](#_Toc195871085)

[**2.** **Définition d’une image :** 5](#_Toc195871086)

[**3.** **Codage d’une image (image numérique) :** 5](#_Toc195871087)

[**4.** **Présentation de Projet :** 5](#_Toc195871089)

[**5.** **Namespace version 1.1 :** 6](#_Toc195871090)

[5.1. Définition : 6](#_Toc195871096)

[5.2. Implémentation des fonctions : 6](#_Toc195871097)

[5.3. Allocation d’images 6](#_Toc195871098)

[5.4. Création d'images particulières : 6](#_Toc195871100)

[5.5. Lecture et écriture de fichiers images au format brut (.raw) : 7](#_Toc195871104)

[5.6. Conversion d'image d'un type en un autre : 8](#_Toc195871106)

[5.7. Conversion d'une image en niveaux de gris en image en fausses couleurs à l'aide d'une LUT : 8](#_Toc195871108)

[5.8. Tester des filtres LUT sur les différentes images : 9](#_Toc195871112)

[**6.** **Namespace version 1.1 :** 9](#_Toc195871113)

[**7.** **Namespace version 2.0 :** 9](#_Toc195871114)

[**8.** **Conclusion :** 9](#_Toc195871115)

Liste des figures

[Figure 1: Représentation d’image numérique. 5](#_Toc195871156)

[Figure 2: Fonction d’allocation d’image. 6](#_Toc195871157)

[Figure 3: Création d’image blanche. 6](#_Toc195871158)

[Figure 4: Création d’image damier. 6](#_Toc195871159)

[Figure 5: Création d’image sinusoïdale. 7](#_Toc195871160)

[Figure 6: Lecteur d’une image .Raw 8](#_Toc195871161)

[Figure 7: Conversion d’une image d’un type à un autre. 8](#_Toc195871162)

[Figure 8: Chargement de fichier LUT. 8](#_Toc195871163)

[Figure 9: Application de LUT sur l’image. 9](#_Toc195871164)

Liste des tableaux

## **Introduction :**

Le traitement d’images, sous-domaine du traitement du signal, regroupe l’ensemble des méthodes et techniques appliquées aux images et vidéos dans le but d’en extraire des informations pertinentes ou d’en améliorer la perception visuelle. Avant toute phase de traitement, un prétraitement est souvent nécessaire afin d’optimiser la qualité des images. Cela inclut, par exemple, des opérations de rehaussement de contraste, de suppression de bruit, de correction du flou, ainsi que des techniques de segmentation ou d’extraction de contours visant à isoler les éléments significatifs d’une image.

Ce rapport présente une synthèse des différentes versions évolutives de notre bibliothèque de traitement d’images en C++. La version de base de notre projet repose sur une architecture fonctionnelle utilisant des **templates**, tandis que les versions suivantes introduisent une approche orientée objet, en utilisant deux classes Image et **ImageRGB** et des surcharges d’opérateurs

La dernière étape de notre projet consiste à implémenter des méthodes de prétraitement (rehaussement de contraste, suppression de bruit, correction du flou), ainsi que des techniques de segmentation et d’extraction de contours, sur différentes images.

## **Définition d’une image :**

Une image, c’est une représentation visuelle d’une personne ou d’un objet, réalisée par des moyens comme la peinture, le dessin, la photo ou la vidéo. C’est aussi un ensemble organisé d’informations qui, une fois affichées à l’écran, forment quelque chose que l’œil humain peut reconnaître.

D’un point de vue mathématique, une image peut être vue comme une fonction en deux dimensions, notée f(x, y), où chaque point (x, y) correspond à une certaine valeur. Cette valeur représente la lumière ou le niveau de gris à cet endroit précis de l’image.

## **Codage d’une image (image numérique) :**

Une image numérique est une représentation visuelle d’un objet ou d’une scène, composée d’une grille de petits éléments appelés pixels. Chaque pixel contient une information, comme une couleur ou un niveau de gris, qui permet de reconstituer l’image dans son ensemble.

Cette image peut provenir d’une photo, d’un dessin ou d’une vidéo, et devient numérique grâce à un processus appelé numérisation, qui convertit une image réelle (analogique) en une matrice de valeurs numériques.

Mathématiquement, une image numérique est représentée par une fonction à deux dimensions, **f(x, y)**, où chaque **point (x, y)** correspond à un pixel, et **f(x, y)** indique l’intensité lumineuse à ce point. Cela signifie que chaque pixel est une mesure de la lumière captée à un endroit précis de l’image.



Figure 1: Représentation d’image numérique.

## **Présentation de Projet :**

Le projet a été conçu pour faciliter et structurer le code destiné au traitement d'images avec C++. Il est composé de plusieurs blocs de type **namespace** afin d'éviter les conflits de noms de fonctions utilisés. Ce projet est organisé en trois blocs **namespace** : **v1.0**, **v1.1** et **v2.0**.

## **Namespace version 1.1 :**



### Définition :

La version 1.0 utilise des fonctions de base pour travailler avec des images. Elle permet de créer différentes images comme des images blanches, des damiers ou encore des images sinusoïdales. Elle peut aussi lire et enregistrer des fichiers d’image au format brut (. Raw). En plus, elle peut transformer des images de haute précision, comme celles en 16 bits ou plus, en images plus simples de 8 bits. Elle permet aussi d’appliquer des filtres de couleur LUT. Toutes ces opérations sont faites en utilisant des tableaux dynamiques **std::vector** en C++.

### Implémentation des fonctions :

### Allocation d’images

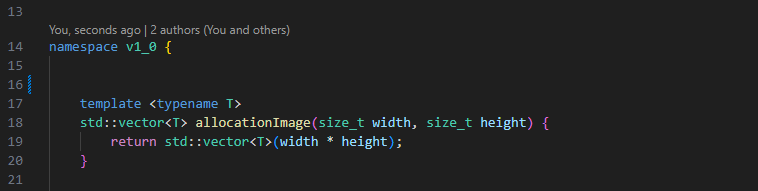


Figure 2: Fonction d’allocation d’image.

Cette fonction sert à allouer dynamiquement une image sous forme à une seule dimension dont la taille correspond à la largeur multipliée par la hauteur de l’image. C’est une fonction de type générique (Template), elle peut fonctionner avec n’importe quel type de données (int, float, unsigned char, etc.).

### Création d'images particulières :

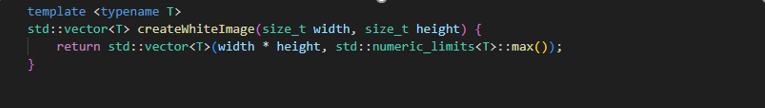
* **Image blanche :**

Figure 3: Création d’image blanche.

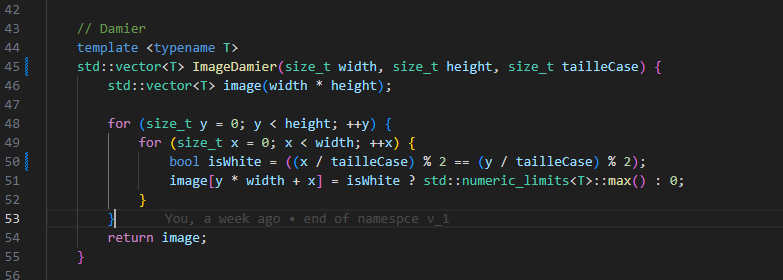
* **Image Damier**

Figure 4: Création d’image damier.

* **Image Sinusoïdale**

Figure 5: Création d’image sinusoïdale.

### Lecture et écriture de fichiers images au format brut (.raw) :

Figure 6: Lecteur d’une image .Raw

### Conversion d'image d'un type en un autre :

Figure 7: Conversion d’une image d’un type à un autre.

### Conversion d'une image en niveaux de gris en image en fausses couleurs à l'aide d'une LUT :

* **Chargement de fichier LUT (le filtre LUT) :**

Figure 8: Chargement de fichier LUT.

* **Application de fausse couleur à l’aide de LUT**

Figure 9: Application de LUT sur l’image.

### Tester des filtres LUT sur les différentes images :

## **Namespace version 1.1 :**

## **Namespace version 2.0 :**

## **Conclusion :**