Ecole Publique d'ingénieures et d'ingénieurs en 3 ans

Rapport

Projet Fondements de l'informatique

Janvier 2024,
version finale

Ayoub Goubraim
Mohamed Ben Lboukht

Ayoub.goubraim@ecole.ensicaen.fr
Mohamed.ben-lboukht@ecole.ensicaen.fr

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

www.ensicaen.fr

TABLE DES MATIERES

INTRODUCTION

1-1-CONTEXTE DU PROJET

A la fin du 5eme semestre, nous avons réalisé un projet qui a mis en valeur nos compétences récemment acquises, notamment dans les domaines des structures de données et de l'allocation Mémoire. Ce projet démontre notre capacité à appliquer pratiquement les concepts abordés en cours, validant notre expertise dans la résolution de problèmes spécifiques en langage C.

1-2-OBJECTIFS DU PROJETS

Le projet vise à développer une technique pour inverser des tables de couleur, nécessitant une image de 24 bits et une table de couleur en entrée. En sortie, cette méthode crée une image dans laquelle chaque pixel est assigné à sa couleur la plus proche dans la table. Ainsi, dans cette initiative, deux approches seront employées pour déterminer la couleur la plus proche pour chaque pixel de l'image. La première méthode, bien que simple, est relativement lente, tandis que la seconde utilise les kd-arbres pour une exécution plus rapide. Ainsi, l'objectif de ce rapport est d'expliciter le processus de création des deux méthodes mentionnées et de fournir une comparaison détaillée de leurs résultats respectifs.

1-3-ORGANISATION DU TRAVAIL

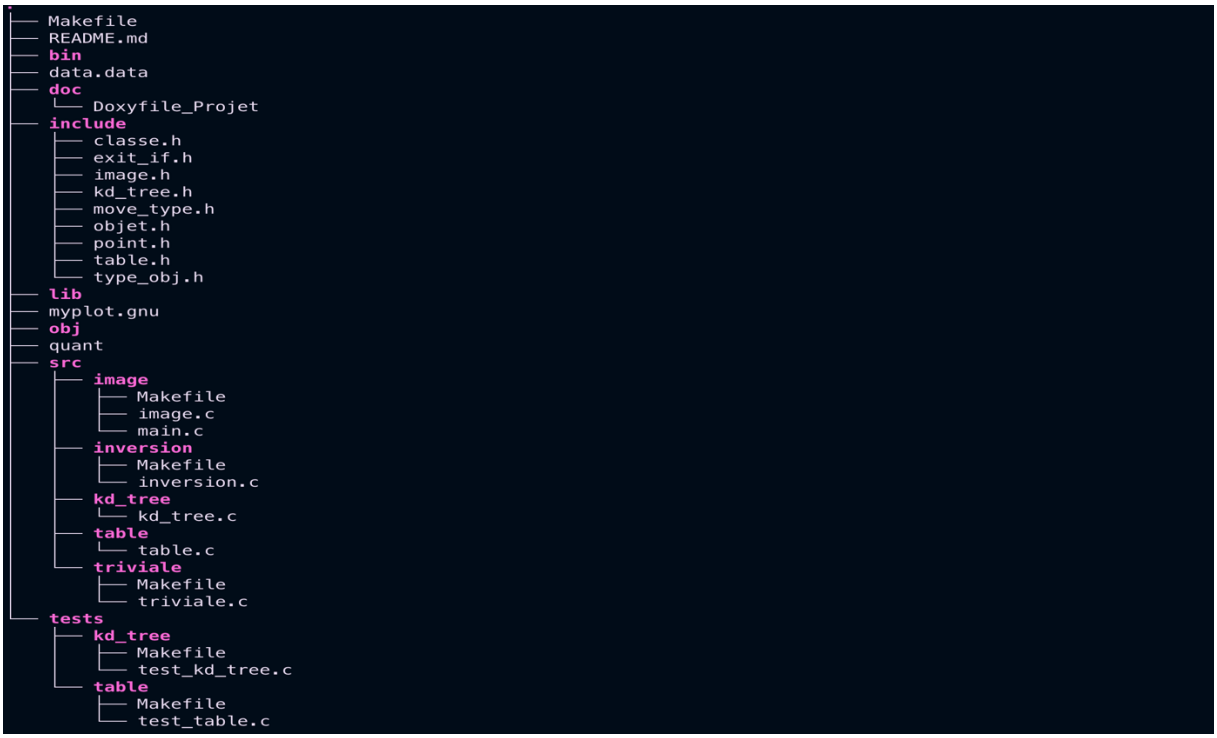

Figure 1: Structure du répertoire projet

MODULE TABLE DE COULEUR

Une image-table de couleurs est définie comme une image avec une hauteur d'une unité multipliée par le nombre de couleurs en largeur. Ce module a pour objectif de créer une structure, "color_table", qui servira à stocker les informations de cette image-table.

2-1-STRUCTURE DE COULEUR TABLE

la structure "color_table" est composée des éléments suivants :

- size: Indique le nombre de couleurs dans la table.
- data: Une table unidimensionnelle contenant les couleurs, avec les composants RGB de chaque couleur.
- owner: Un booléen qui indique si la table est une sous-table ou non.

La stratégie adoptée consiste à utiliser un tableau unidimensionnel pour aligner les pixels de manière contiguë, bien que cela entraîne une taille trois fois plus grande en raison de la disposition en blocs de 3 par 3. On a écarté l'utilisation de tableaux de pointeurs pour éviter une multiplication excessive des appels à malloc et privilégier une optimisation de la gestion de la mémoire.

2-2-FONCTIONS

- La fonction **create_color_table** alloue la mémoire et génère une table de couleurs à partir d'une image. Elle parcourt les pixels, ajoute les composantes RGB à la table, fixe la propriété "owner" à vrai, puis renvoie la table créée.
- La fonction **destroy_color_table** libère la mémoire du tableau de couleurs et de la structure associée. La libération du tableau se fait uniquement si la table n'est pas une sous-table pour éviter des altérations potentielles.
- La fonction **color_table_duplicate** crée une copie d'une partie spécifiée d'une table de couleurs en allouant la mémoire nécessaire. La nouvelle table a une taille définie et la propriété "owner" est initialisée à 0 (False).
- La fonction **color_table_get_color** récupère la couleur d'une table de couleurs à un indice spécifié.
- La fonction **color_table_get_nb_color** retourne le nombre de couleurs dans une table de couleurs donnée

- La fonction **color_table_sort** trie une table de couleurs en fonction de l'axe spécifié. Elle utilise le tri à bulles pour trier les pixels de la table en fonction de l'axe choisi.

MODULE KD TREE

Un arbre kd <k-dimensional> est une structure de données arborescente utilisée dans le domaine de l'informatique pour organiser des points dans un espace multidimensionnel. Il est souvent employé pour des tâches telles que la recherche spatiale, notamment dans les applications de traitement d'images, de reconnaissance de formes, et de bases de données géospatiales.

3-1-STRUCTURE DE KD TREE

La structure "kd_tree" est définie avec les éléments suivants :

- table: Une table de couleurs représentant le kdtree.
- left_son: Un pointeur vers le sous-arbre gauche du kdtree.
- right_son: Un pointeur vers le sous-arbre droit du kdtree.
- split_axis: L'axe sur lequel le kdtree est divisé.
- split_plan: La valeur du plan de division du kdtree.

L'arbre kd est construit récursivement, où la racine contient la table complète. Chaque nœud divise la table en deux parties triées, avec le fils gauche contenant la moitié inférieure et le fils droit la moitié supérieure. Cette division se poursuit jusqu'à ce que chaque feuille contienne une portion minime de la table initiale.

3-2-FONCTIONS

- La fonction **choose_axis** détermine l'axe de division optimal (red, green, ou blue) pour un arbre kd en évaluant les plages de valeurs des composantes RGB dans la table de couleurs, basé sur le tri de chaque composante.
- La fonction **choose_plan** calcule le plan de division optimal pour un arbre kd à partir d'une table triée selon un axe. Le nombre de couleurs inférieur au plan est renvoyé via le pointeur number_color.
- La fonction **create_kdtree** construit récursivement un arbre kd à partir d'une table de couleurs. Elle commence par allouer dynamiquement l'espace mémoire pour la structure de l'arbre et la table associée. Si la taille de la table dépasse la valeur maximale (max), elle trie la table de couleurs à l'aide de la fonction **color_table_sort**. Ensuite, elle

détermine un axe de division optimal et un plan de division en utilisant les fonctions **choose_axis** et **choose_plan**. Elle divise ensuite la table en deux sous-tables triées selon le plan de division et crée récursivement les sous-arbres gauche et droit. Si la taille est inférieure à la valeur maximale, les sous-arbres sont définis comme nuls. Finalement, la fonction renvoie l'arbre kd ainsi construit.

- La fonction **destroy_kdtree** libère la mémoire d'un arbre kd de manière récursive. Elle commence par détruire le sous-arbre gauche, puis le sous-arbre droit, et enfin elle libère la mémoire de la table de couleurs associée au nœud actuel, suivie par la libération du nœud lui-même.

- La fonction **distance** calcule la distance euclidienne entre deux couleurs via leurs composantes RGB

- La fonction **closest_color** identifie la couleur la plus proche d'une couleur de référence dans une table donnée, en calculant les distances euclidiennes et en mettant à jour la couleur la plus proche à chaque itération. Des assertions garantissent que les entrées ne sont pas nulles.

- La fonction **kdtree_get_nearest_color** explore de manière récursive l'arbre kd pour trouver la feuille contenant la couleur la plus proche dans une table de couleurs par rapport à une couleur donnée dans une image. La recherche descend dans l'arbre en fonction du plan de coupe de chaque nœud, utilisant ensuite la fonction **nearest_color** pour trouver la couleur la plus proche dans la partie spécifique à cette feuille. En remontant dans l'arbre, elle vérifie que la distance entre la couleur recherchée et la couleur trouvée est inférieure à la distance entre la couleur recherchée et le plan de coupe. Si ce n'est pas le cas, une nouvelle recherche est effectuée dans la table spécifique à l'autre feuille non explorée, calculant une nouvelle distance entre cette nouvelle couleur et la couleur de référence. Elle compare ensuite cette nouvelle distance à l'ancienne pour garantir la précision de la recherche de la couleur la plus proche.

INVERSION

4-1-PAR LA METHODE TRIVIALE

4-1-1-DESCREPTION

Le programme débute par le chargement d'une image et d'une table de couleurs, suivie de la création d'une table de couleurs à partir de cette dernière à l'aide de la fonction color_table_create. Pour l'inversion de l'image, chaque pixel est parcouru à l'aide des fonctions image_debut et image_pixel_suivant. Pour chaque pixel, la couleur la plus proche est recherchée dans la table de couleurs en calculant la distance euclidienne. La couleur du pixel est ensuite remplacée par

celle de la table ayant la distance la plus faible. Enfin, l'image résultante est sauvegardée grâce à la fonction image_sauvegarder du module image.

4-1-1-LES AVANTAGES ET LES INCOVENIENTS

L'avantage de cette méthode réside dans sa simplicité, car la recherche de la couleur la plus proche se fait directement en parcourant la table de couleurs sans nécessiter de structure supplémentaire. Cependant, l'inconvénient majeur est sa lenteur, car la recherche est effectuée de manière séquentielle pour chaque pixel, ce qui peut entraîner des performances inefficaces, surtout avec de grandes tables de couleurs.

4-1-PAR LA METHODE KD TREE

4-1-1-DESCREPTION

Lors de l'inversion par kd-arbre, l'algorithme reste similaire à celui de la méthode triviale, à la différence que lors de la recherche de la couleur la plus proche dans la table de couleurs pour chaque pixel de l'image, on utilise la fonction kdtree_get_nearest_color.

4-1-1-LES AVANTAGES ET LES INCOVENIENTS

Cette fonction exploite la structure de l'arbre kd pour effectuer une recherche plus efficace de la couleur la plus proche. Au lieu de parcourir linéairement la table de couleurs, elle utilise l'arbre kd pour explorer sélectivement les branches les plus prometteuses, réduisant ainsi le nombre de comparaisons nécessaires et améliorant les performances globales de l'inversion des couleurs. Cela rend le processus plus rapide, en particulier avec des tables de couleurs de grande taille.

LES TESTS

5-1-PAR LA METHODE TRIVIALE

Figure 2 : image Fille avant/apres inversion par la table zelda 1280.

5-1-PAR LA METHODE KD TREE

Figure 3 : image Zelda avant/apres inversion par la table house 64.

CONCLUSION

L'utilisation de la méthode kd-arbre garantit une nette amélioration des temps d'exécution lors de l'inversion des couleurs, notamment pour les tables comportant un grand nombre de couleurs. Cette différence est particulièrement notable lors de l'inversion d'une table contenant des couleurs proches de l'image. Des courbes ont été générées pour des tables de différentes tailles (8, 16, 32, 64, 128, 256, 512, 768, 1280, 2048) à l'aide de Gnuplot, en utilisant le script myplot.gnu et le fichier data.data.

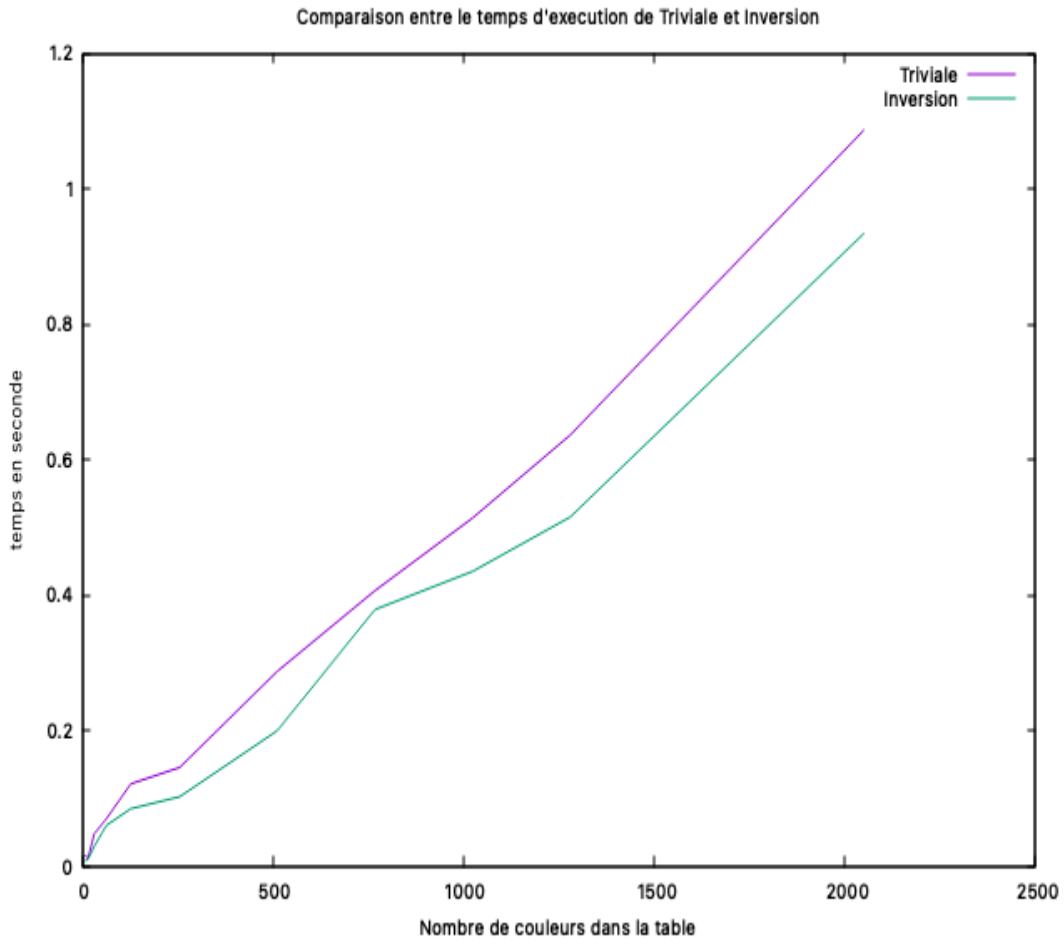

Figure 4 : Temps d'execution d'une table provenant d'une image differente.

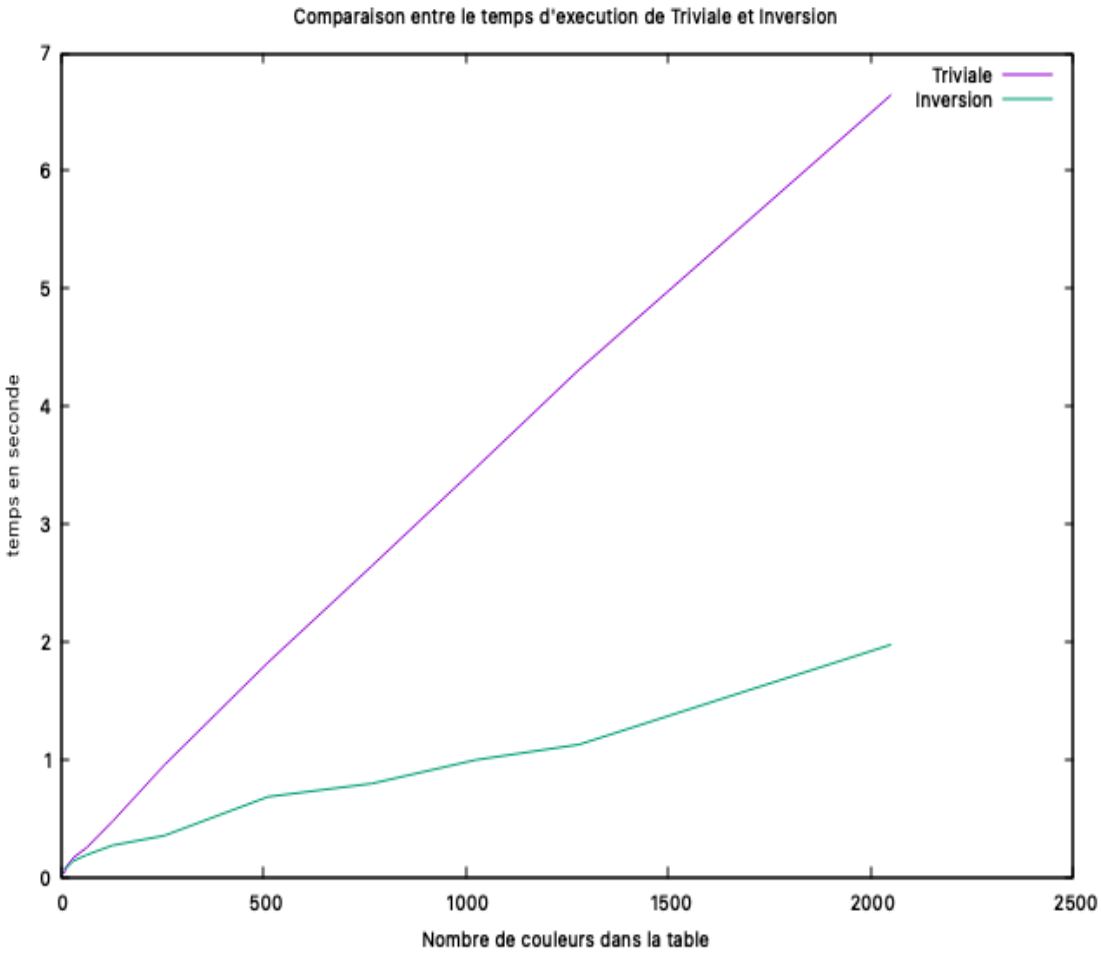

Figure 5 : Temps d'execution d'une table provenant de cette image.

.

ENSI CAEN

ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Ecole Publique d'ingénieures et d'ingénieurs en 3 ans

6 boulevard Maréchal Juin, CS 45053
14050 CAEN cedex 04

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

RÉGION NORMANDIE

Normandie Université

Cti
Commission des titres d'ingénieur

European Accreditation of Engineering Programmes
EUR-ACE®

CONFÉRENCE DES GRANDES ÉCOLES

cdefi
Conférence des Directeurs des Écoles Françaises d'Ingénieurs

INP
CONCOURS COMMUN INP