HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Alberto Dassatti, Bruno
Da Rocha Carvalho

Laboratoire de High Performance Coding

semestre printemps 2024

Laboratoire 4 : SIMD

Temps à disposition: 6 périodes (3 séances de laboratoire)

1 Objectifs de ce laboratoire

Dans ce laboratoire, vous serez amené(e)s à analyser et à écrire du code en utilisant les instructions SIMD. Vous trouverez toutes les informations nécessaires concernant les instructions SIMD pour les processeurs Intel à l'adresse suivante : https://www.intel.com/content/www/us/en/docs/intrinsics-guide/index.html

2 Analyse et amélioration

Dans cette première partie du laboratoire, vous devrez analyser et modifier le code présent dans `smid_lab/src/k-means.c` et `smid_lab/include/k-means.h` en y intégrant les optimisations SIMD que vous jugerez pertinentes.

Afin de compiler et exécuter le projet, un cmake vous est fournit que vous pouvez utilisez comme :

```
// Depuis le répertoire ou le CMake est.
cmake -B build // Permet de créer un dossier build avec les config CMake
cmake --build build // Permet de build le projet

// Vous pouvez clean avec :
cmake --build build --target clean
```

Dans les sections 2.1 et 2.2, vous trouverez des explications sur les algorithmes mis en place et leurs objectifs.

2.1 Segmentation d'image

La segmentation d'image est un processus consistant à diviser une image en plusieurs parties ou segments qui représentent des zones distinctes de l'image. Cette technique est largement utilisée en traitement d'images et en vision par ordinateur pour extraire des informations importantes à partir d'images.

Cette méthode est largement utilisée dans de nombreux domaines, tels que la reconnaissance de forme, la surveillance de la circulation, la médecine, l'analyse de données géologiques, l'astronomie, etc. Elle est également utilisée dans des applications pratiques telles que la reconnaissance de visages et la détection d'objets dans les images de caméras de sécurité.

Vous trouverez ci-dessous un exemple d'exécution du programme de segmentation d'image fourni, avec deux clusters :

Figure 1: Image de base

Figure 2: Image ségmentée

2.2 K-Means et K-means++

K-means est un algorithme de clustering utilisé pour regrouper des données en k groupes distincts. L'algorithme utilise une technique itérative pour déplacer les centres des clusters jusqu'à ce que la somme des carrés des distances entre les points et leur centre de cluster le plus proche soit minimisée. L'algorithme k-means est simple et rapide, mais il est sensible à l'initialisation des centres de clusters et peut être piégé dans des minima locaux.

Pour améliorer les performances de k-means, une technique d'initialisation des centres de cluster appelée k-means++ a été proposée. K-means++ utilise une méthode de sélection des centres de cluster qui garantit que les centres de cluster initiaux sont répartis uniformément dans l'espace de données. En conséquence, les centres de cluster initiaux sont plus susceptibles de se rapprocher de la solution globale optimale. K-means++ est une technique simple mais puissante pour améliorer les performances de l'algorithme k-means.

Dans le cadre de ce laboratoire, K-means est utilisé pour segmenter des images en k groupes, en se basant sur la teinte de chaque pixel. Ainsi, chaque pixel de l'image est assigné au cluster ayant la couleur RGB la plus proche. Cette méthode permet de créer des groupes d'images similaires en termes de couleur.

2.3 Code fournit

Le code qui vous est proposé se compose comme suit :

- Un fichier main.c qui charge l'image, appelle la fonction de segmentation et sauvegarde l'image modifiée dans un fichier de résultat.
- Une fonction de distance, qui retourne la distance euclidienne entre deux pixels RGB.

```
float distance(uint8_t* p1, uint8_t* p2);
```

- Les deux fonctions de k-means et de k-means++.

```
void kmeans_pp(struct img_1D_t *image, int num_clusters, uint8_t *centers);

void kmeans(struct img_1D_t *image, int num_clusters);
```

Bien sûr, tout ce code n'est qu'une proposition. Il est là pour vous assurer du résultat à obtenir et pour vous permettre de faire vos tests facilement. Vous êtes libre de le modifier à votre guise, tant que le fichier main.c reste intact.

2.4 Contraintes et conseils

- Vous ne devez pas modifier le fichier `main.c`.
- Si vous souhaitez utiliser et tester d'autres images (plus grandes ou plus complexes), elles doivent être au format png.

- L'utilisation de l'exécutable se fait comme suit : `./build/segmentation <img_src.png> <nb_clusters> <img_dest.png>`.
- Vous devrez modifier le CMake pour pouvoir gérer les instructions SIMD.
- Vous êtes autorisé à ajouter, supprimer ou modifier tout ce que vous voulez dans le code de `k-means.c/.h`.
- Vous ne pouvez pas utilisé des vecteurs plus grand que 256 bits, soit AVX2.

3 Développement et réflexion sur l'utilisation SIMD

Au cours d'un exercice précédent, vous avez implémenté un algorithme de détection de contours. Nous vous demandons maintenant de reprendre votre code, de l'intégrer dans le projet de ce laboratoire (le CMakeLists est déjà prêt à compiler ce projet) et de l'analyser afin de l'optimiser à l'aide d'instructions SIMD ou d'autres techniques vues dans le cours.

Si vous avez créé d'autres fichiers pour les laboratoires précédents, ajoutez-les au fichier CMakeLists.txt.

4 Travail à rendre

Pour les deux parties du laboratoire, il vous sera demandé de rendre votre code modifié, compilable et accompagné d'un rapport d'analyse. Vous devrez expliquer vos choix et présenter les avantages obtenus grâce à ces changements. Par exemple, vous pourrez expliquer comment vous avez optimisé une opération pour traiter davantage de données simultanément, ou démontrer en quoi vos modifications améliorent la performance globale du code.

Afin de motiver les étudiants à accomplir le travail sur le code de segmentation d'image, un bonus de +1 sera accordé à celui dont le code sera le plus rapide. Les tests de performance seront réalisés sur une machine de laboratoire en utilisant simplement la fonction 'time', il n'est donc pas nécessaire d'ajouter des mesures de temps dans votre code. Ces tests seront effectués sur plusieurs images et la moyenne des temps sera prise en compte pour déterminer le gagnant.