

SIMD and NEON intrinsics

TIC

unité d'enseignement CNM

Auteurs: **Andrè Costa & Alexandre Iorio**

Professeur: **Marina Zapater**

Assistant: **Mehdi Akeddar**

Salle de laboratoire **A09**

Date: **10.11.2024**

Table des matières

- 0. Introduction
- 1. Stage 2 - détection de contours
- 2. Analyse des performances
- 3. Conclusion
- 4. Réf

0. Introduction

Dans ce laboratoire, nous allons appliquer du calcul matriciel en utilisant les instructions SIMD et NEON. Nous allons comparer les performances de calcul avec les laboratoires précédents.

1. Stage 2 - détection de contours





Dans cette partie, nous allons implémenter la détection de contours en utilisant les instructions SIMD et NEON et




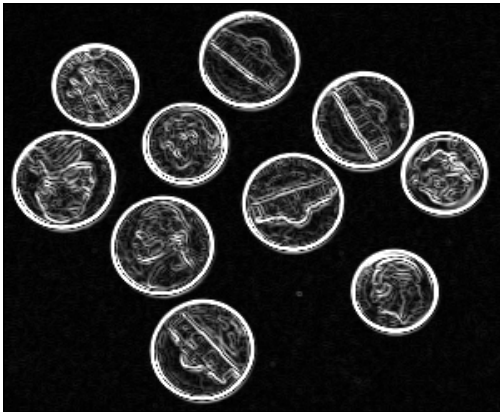
Afin de pouvoir utiliser les vecteurs de `int16x8_t` utilisé comme kernel 3×3 , il faut admettre la valeur du centre du kernel à 0.

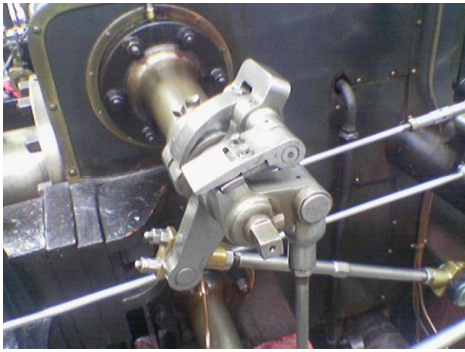
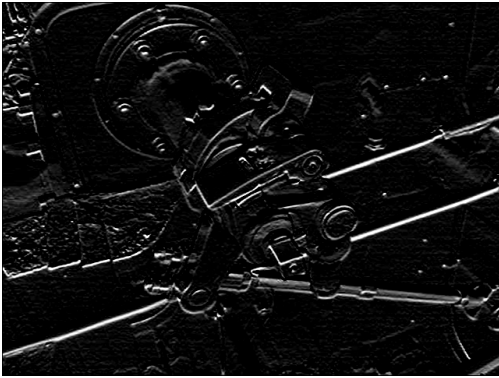

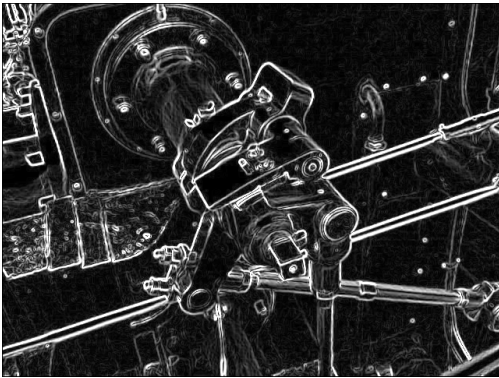
Le principe est que pour chaque itération du kernel, on va créer un nouveau vecteur de `int16x8_t` avec les valeurs de l'image où se trouve le kernel pour pouvoir les multiplier en une opération et les accumuler dans un tableau 2D de la taille de l'image.

Les valeurs seront ensuite normalisées entre 0 et 255 pour pouvoir les afficher en tant qu'image de nuances de gris.

Voici le résultat de la détection de contours avec les instructions SIMD et NEON:

Originale	Détection de bords horizontaux
	
Détection de bords verticaux	Détection de contours
	

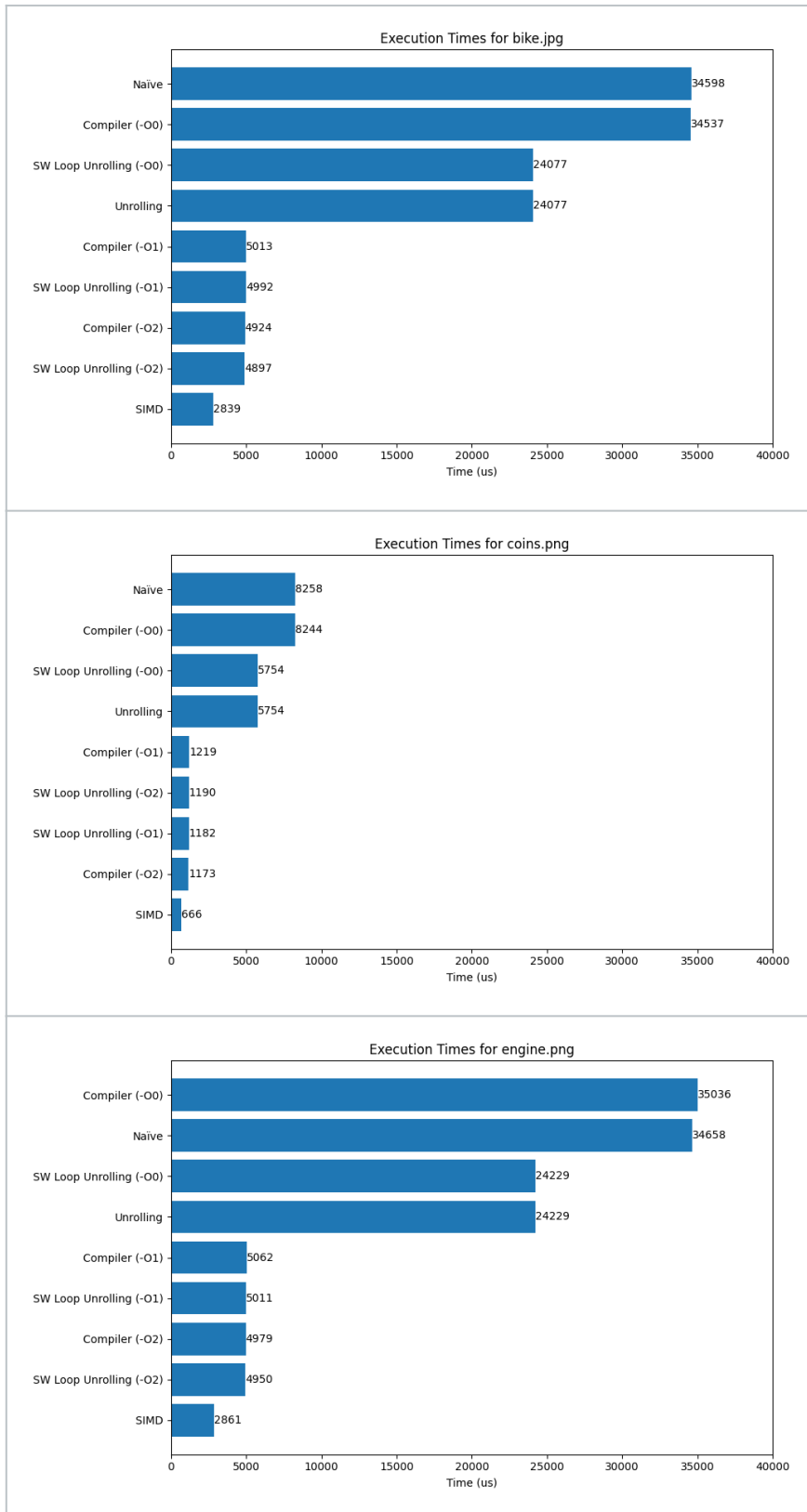
Originale	Détection de bords horizontaux
	
Détection de bords verticaux	Détection de contours
	

Originale	Détection de bords horizontaux
	
Détection de bords verticaux	Détection de contours
	

2. Analyse des performances

Afin de comparer les performances de calcul entre les différentes implémentations des laboratoires précédents et l'implémentation SIMD.

Voici les résultats obtenus:



On remarque que l'implémentation SIMD est plus rapide que les implémentations précédentes.

3. Conclusion

Dans ce laboratoire, nous avons implémenté la détection de contours en utilisant les instructions SIMD et NEON. Nous avons comparé les performances de calcul avec les laboratoires précédents et nous avons constaté que l'implémentation SIMD est plus rapide. Ces résultats nous ont étonné car nous pensions pas gagner autant de temps. En effet, la plus grande perte de temps se situe au niveau des accès memoires. On se rend bien compte que le nombre d'instruction à un rôle prépondérant dans la performance d'un code.

4. Réf

- ChatGPT pour la génération du script `chart.py`