Índice

1. Introducción

2. Rotar

2.1. Código C

2.2. Código Asm

2.3. Experimentación

2.3.1. Idea

2.3.2. Hipótesis

2.3.3. Resultados

3. Smalltiles

3.1. Código C

3.2. Código Asm

3.3. Experimentación

3.3.1. Idea

3.3.2. Hipótesis

3.3.3. Resultados

4. Pixelar

4.1. Código C

4.2. Código Asm

4.3. Experimentación

4.3.1. Idea

4.3.2. Hipótesis

4.3.3. Resultados

5. Combinar

Este filtro permite reutilizar cálculos debido a dos factores. Uno de ellos es la forma que tiene cada píxel generado en la imagen resultante, que consiste en que si se tienen una imagen A y una imagen B de tamaño $m \times n$ entonces el píxel de la imagen generada ${I_{AB}}^{i,j}$ se calcula de la siguiente forma:

$$ {I_{AB}}^{i,j} = \frac{\alpha * ({I_A}^{i,j} - {I_B}^{i,j})}{255{,}0} + {I_B}^{i,j} $$

El otro factor es que nuestro filtro está optimizado para casos en los que la imagen B es el reflejo vertical de la imagen A. Es decir que se da que ${I_A}^{i,j} = {I_B}^{i,n-j+1}$ como se puede apreciar en la siguiente figura.

Figura 1: Se muestra la imagen A del lado izquierdo con la imagen B, el reflejo vertical de A, en la parte derecha.

Entonces se tiene que

$$ \begin{aligned} {I_{AB}}^{i,j} &= \frac{\alpha * ({I_A}^{i,j} - {I_B}^{i,j})}{255{,}0} + {I_B}^{i,j} \text{ por la fórmula del filtro} \\ &= \frac{\alpha * ({I_A}^{i,j} - {I_A}^{i,n-j+1})}{255{,}0} + {I_A}^{i,n-j+1} \text{ dado que} {I_A}^{i,j} = {I_B}^{i,n-j+1} \end{aligned} \tag{1} $$

y análogamente

$$ \begin{aligned} {I_{AB}}^{i,n-j+1} &= \frac{\alpha * ({I_A}^{i,n-j+1} - {I_B}^{i,n-j+1})}{255{,}0} + {I_B}^{i,n-j+1} \\ &= \frac{\alpha * ({I_A}^{i,n-j+1} - {I_A}^{i,j})}{255{,}0} + {I_A}^{i,j} \end{aligned} \tag{2} $$

Así, se obtiene

$$ \begin{aligned} {I_{AB}}^{i,n-j+1} &= -1 * \frac{\alpha * [-1 * ({I_A}^{i,n-j+1} - {I_A}^{i,j})]}{255{,}0} - {I_A}^{i,n-j+1} + {I_A}^{i,n-j+1} + {I_A}^{i,j} \\ &= -1 * \left[\frac{\alpha * ({I_A}^{i,j} - {I_A}^{i,n-j+1})}{255{,}0} + {I_A}^{i,n-j+1}\right] + {I_A}^{i,n-j+1} + {I_A}^{i,j} \\ &= -1 * {I_{AB}}^{i,j} + {I_A}^{i,n-j+1} + {I_A}^{i,j} \text{usando la igualdad de (2)} \end{aligned} \tag{3} $$

Estos cálculos muestran que luego de hacer el procesamiento para generar un píxel de la parte izquierda de la imagen resultante se puede obtener el píxel que corresponde a la mitad derecha con pocos cálculos más. Más aún, si se denomina

$$ P = \frac{\alpha * ({I_A}^{i,j} - {I_A}^{i,n-j+1})}{255{,}0} $$

se consigue

$$ {I_{AB}}^{i,j} = P + {I_A}^{i,n-j+1} $$

y

$$ {I_{AB}}^{i,n-j+1} = -P + {I_A}^{i,j} $$

y son menos cálculos necesarios.

5.1. Código C

5.2. Código ASM

5.3. Experimentación

5.3.1. Idea

5.3.2. Hipótesis

5.3.3. Resultados

6. Colorizar

6.1. Código C

El codigo C se trata de una conjuncion de Fors, el exterior que recorre desde la segunda fila hasta la ante ultima, y el interior que recorre desde la segunda columna hasta la ultima, dejando asi afuera a todos los bordes, tal como el enunciado decia. Luego en cada iteracion del ciclo interior, que es donde se hace las opearciones que modifican la imagen, lo que hacemos es crear un arreglo de unsinged chars, res", que es donde guardamos los maximos de cada canal en comparacion a todos los pixeles lindantes del pixel en el cual estemos parado.

- Res [0] ← MaximoLindantesAzul
- Res [1] ← MaximoLindantesVerde
- Res [2] ← MaximoLindantesRojo

Luego con estos tres valores calculamos el alpha correspondiente de cada canal por el cual voy a multiplicar a cada uno. Y por ultimo reescribimos el pixel final, en la imagen source con cada canal multiplicado por dicho alpha.

6.2. Código Asm

El Código en ASM se trata tambien de una conjuncion de ciclos. El recorre las filas desde la segunda hasta la anteUltima, y el interior recorre las columnas desde la segunda hasta la anteultima, pero saltando de a dos pixeles, que es la cantidad que procesamos simultaneamente con instruccciones SSE. El ciclo interior consta de tres partes, la primera es tan pronto se levanta de memoria los pixeles a procesar y todos sus lindantes, calculamos en dos registros los maximos de cada canal, de ambos pixeles a procesar con respecto a todos sus lindantes, y los guardamos ambos en un registro xmm. Luego la segunda parte es calcular el maximo de los maximos de ambos al mismo tiempo. Luego atravez de una par de operaciones, muy especificas para explicar, logro tener un registro xmm de dw, con cada dw representando un canal de cada pixel, en el orden establecido (argb), donde tengo un 1-alpha en la posicion del canal que no teine el maximo de los maximos, y un 1+alpha en la posicion que tiene al maximo de los maximos, y luego concluyo multiplicando a cada pixel por su registro con los alphas indicados, lo reduzco a tamaño de 32 bits por pixel, los uno y los escribo en el destino.

6.3. Experimentación

6.3.1. Idea

En la experimentación de este filtro al igual que en el resto vamos a comparar el rendimiento respecto a los ciclos de clock, que tiene la funcion colorizar en C desde -o0 a -o3 contra asm. Sin embargo luego vamos a contrastar la funcion de asm, contra sigo mismo pero primero agragando jumps de forma que no influya el flujo del programa solo para molestar al jump predictor. Luego vamos a correrlo tal cual esta, y de apoco vamos a ir desenrollando el Código. Como dijimos tiene un ciclo externo y uno interno, por lo que vamos a desenrrollar primero el interno 2 veces, luego 4,16,32 y ver que pasa, y finalmente vamos a saltar a desenrrollarlo completamente, cosa de no dejar casi ninguno controlador de flujo.

6.3.2. Hipótesis

Nuestra hipótesis es que el rendimiento va a ir mejorando a medida que vayamos cambiando los programas respectivamente a como los fui mencionando, osea el mas lento va a ser el Código de asm, molestando al jmp predictor, y el mas rapido el desenrollando el Código 32 veces. El ultimo test, por mas que desenrollemos todo el programa creemos justamente que va a ser el mas lento, por el tamaño del Código, creemos que al hacer un código de tamaño mayor al de la cache, en un momento el pc va a estar haciendo muchisimos mas miss en la cache, que los que ahorra sacando los controladores de flujo.

6.3.3. Resultados

Graficos lindos de lucia :D
ciclos promedio colorizar original 10000 iteraciones: