# MEMORIA TRABAJO EN GRUPO

## Introducción

El trabajo en grupo consiste realizar un programa en C, que utilizando una imagen original, en nuestro caso la imagen "bailarina.bmp", cree una segunda imagen que sea la inversión en blanco y negro de la original, aplicando el siguiente filtro especificado como función matemática:

1. Convert to black & white

$$L_i = 0.3R_i + 0.59G_i + 0.11B_i$$

2. Invert

$$L'_i = 255 - L_i$$

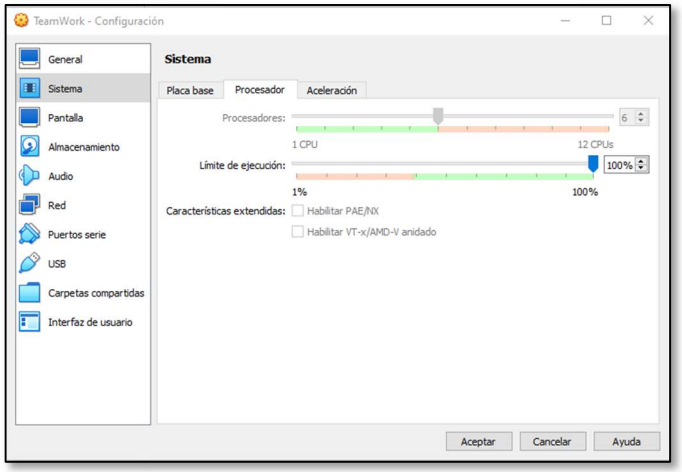
Además, en el enunciado se nos especifica la utilización de tipo de datos "*float*". Y el paquete __256 para la versión SIMD de la fase 2 del proyecto.

Por ello, el desarrollo del trabajo consta de 2 fases:

1. Versión monohilo
2. Versión SIMD y versión multihilo

## Información del sistema

Las mediciones de los programas en sus tres versiones se han realizado con la siguiente configuración y número de procesadores:

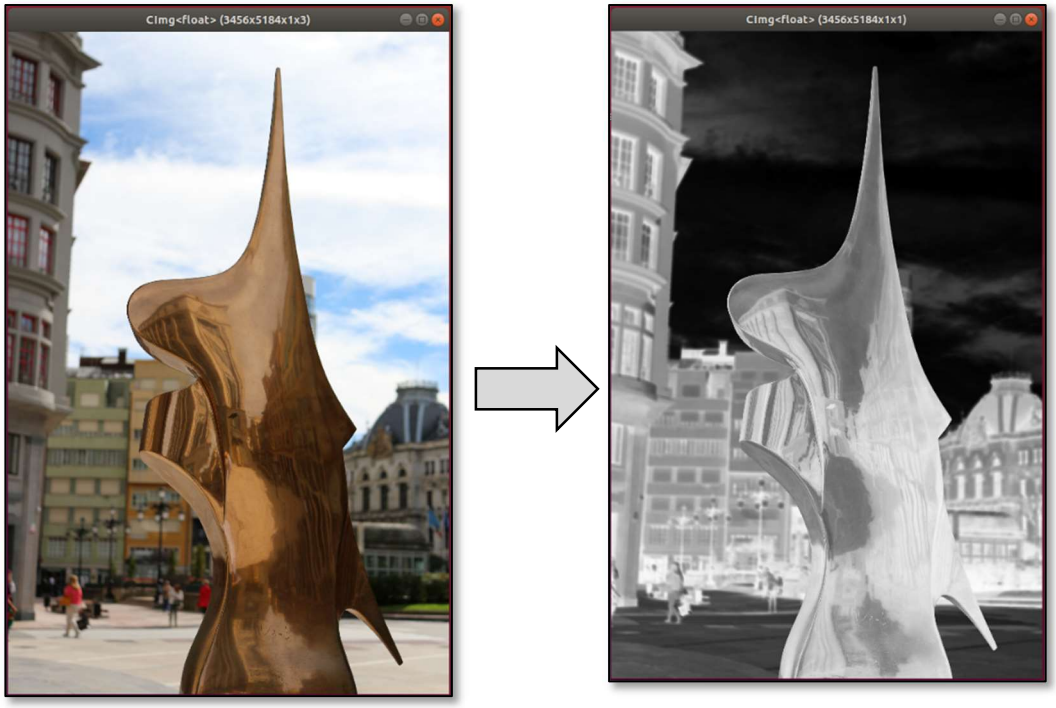





Alejando Medina Fernández – UO288967
César Fernández Tejero – UO284671
Diego Mieres Molinero – UO282386
Gustavo Sobrado Aller – UO286277

## FASE 1: VERSIÓN MONOHILO

En esta primera fase se utiliza un programa monohilo cuyo código está en la carpeta 2023-single-thread-PL07-A del repositorio. Con este programa se aplica el filtro anteriormente especificado para generar la imagen resultante.




Las mediciones obtenidas tras las 10 ejecuciones del programa monohilo son las siguientes:

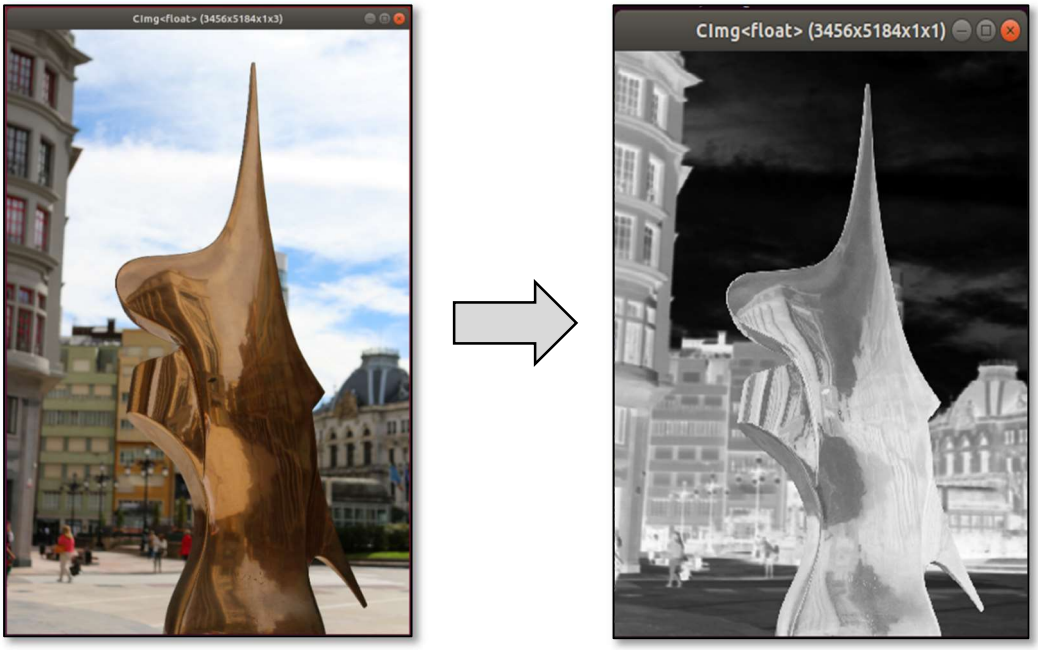
| | t_1 (s) | t_2 (s) | t_3 (s) | t_4 (s) | t_5 (s) | t_6 (s) | t_7 (s) | t_8 (s) | t_9 (s) | t_10 (s) | Media t_1 a t_10 (s) | Desv. Típ. t_1 a t_10 (s) | Intervalo de Confianza de | | Productividad (tareas/min) | Aceleración (SUT/ref) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | Inf. (s) | Sup. (s) | | |
| SingleThread | 6,7142 | 7,3752 | 6,9199 | 6,7712 | 6,8882 | 6,7399 | 7,3783 | 6,9810 | 6,9518 | 6,6222 | 6,9342 | 0,2596 | 6,6118 | 7,2565 | 8,6528 | - |



Alejando Medina Fernández – UO288967
César Fernández Tejero – UO284671
Diego Mieres Molinero – UO282386
Gustavo Sobrado Aller – UO286277

## FASE 2: VERSIÓN SIMD:

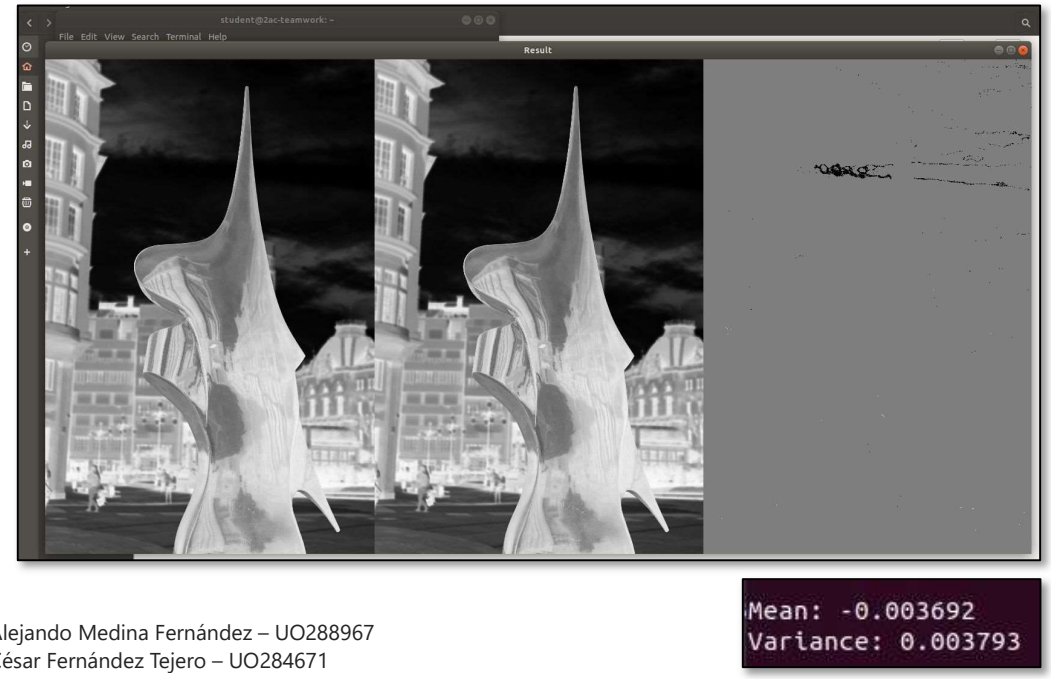
En esta segunda fase se nos propone realizar dos versiones que generen el mismo resultado que la versión monohilo. Primero comenzamos con la versión SIMD, cuyo código se encuentra implementado en el repositorio 2023-simd-PL07-A. Para la realización de este programa debemos tener en cuenta el paquete que se nos especifica en el enunciado: __256. Aplicando el filtro con esta versión generamos la siguiente imagen:




Las mediciones obtenidas tras las 10 ejecuciones del programa con versión SIMD son las siguientes:

| | t_1 (s) | t_2 (s) | t_3 (s) | t_4 (s) | t_5 (s) | t_6 (s) | t_7 (s) | t_8 (s) | t_9 (s) | t_10 (s) | Media t_1 a t_10 (s) | Desv. Típ. t_1 a t_10 (s) | Intervalo de Confianza de Inf. (s) | Sup. (s) | Productividad (tareas/min) | Aceleración (SUT/ref) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SIMD | 1,1301 | 1,1750 | 0,3757 | 1,2421 | 1,3488 | 1,1625 | 1,2613 | 1,1477 | 1,3133 | 1,4505 | 1,1607 | 0,2938 | 0,7959 | 1,5255 | 51,6931 | 5,9741 |

Utilizando la aplicación denominada DiffImages, comprobamos que las imágenes resultado obtenidas con las versiones SIMD y multihilo coinciden con la imagen obtenida como resultado de la fase1.

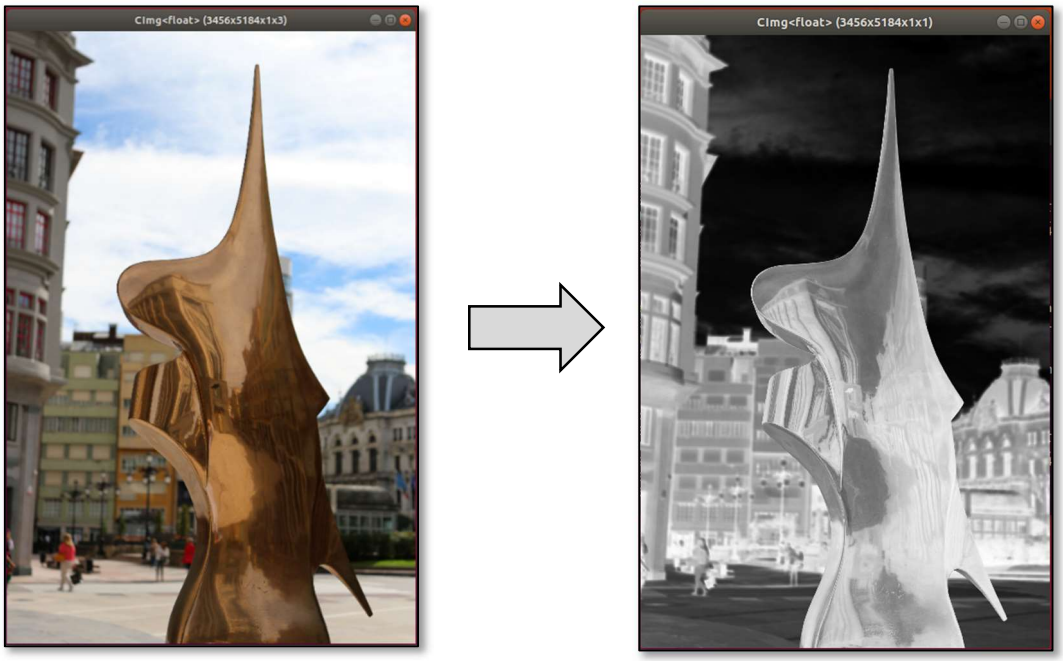





Alejando Medina Fernández – UO288967
César Fernández Tejero – UO284671
Diego Mieres Molinero – UO282386
Gustavo Sobrado Aller – UO286277

## FASE 2: VERSIÓN MULTIHILO:

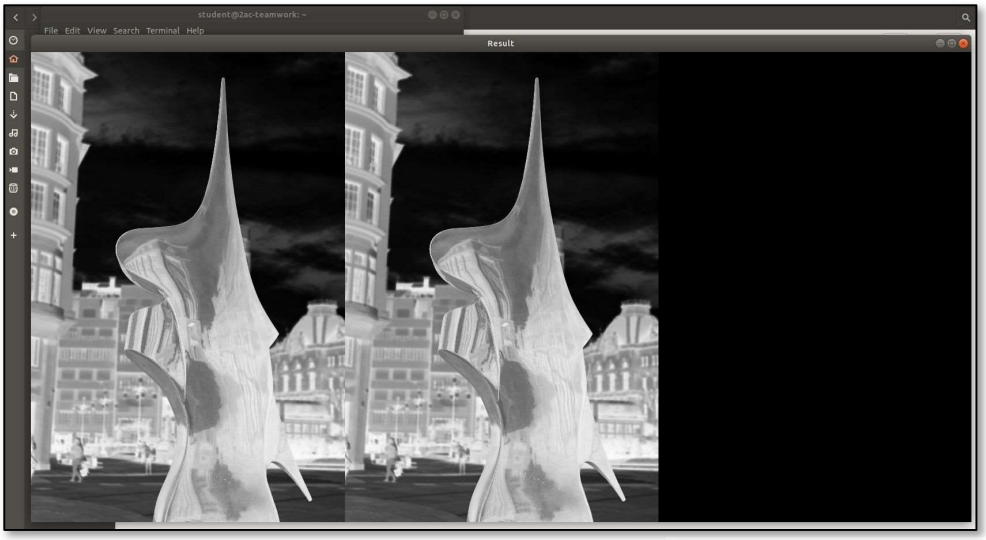
La última parte de esta fase consta de realizar un programa con el mismo objetivo que las versiones anteriores, pero esta vez mediante un sistema multihilo. Para ello, se crea la imagen generada dividiendo las componentes de color entre el número de hilos, de forma que cada hilo solo genera una parte de la imagen. Una vez aplicado el filtro, obtenemos la siguiente imagen resultante:




Las mediciones obtenidas tras las 10 ejecuciones del programa con versión multihilo son las siguientes:

| | t_1 (s) | t_2 (s) | t_3 (s) | t_4 (s) | t_5 (s) | t_6 (s) | t_7 (s) | t_8 (s) | t_9 (s) | t_10 (s) | Media t_1 a t_10 (s) | Desv. Típ. t_1 a t_10 (s) | Intervalo de Confianza de Inf. (s) | Sup. (s) | Productividad (tareas/min) | Aceleración (SUT/ref) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MultiThread | 0,0375 | 0,0379 | 0,0430 | 0,0350 | 0,0393 | 0,0381 | 0,0427 | 0,0452 | 0,0397 | 0,0423 | 0,0401 | 0,0031 | 0,0362 | 0,0439 | 1497,2750 | 173,0394 |

Utilizando la aplicación denominada DiffImages, comprobamos que las imágenes resultado obtenidas con las versiones SIMD y multihilo coinciden con la imagen obtenida como resultado de la fase1.







Alejando Medina Fernández – UO288967
César Fernández Tejero – UO284671
Diego Mieres Molinero – UO282386
Gustavo Sobrado Aller – UO286277

# Carga de trabajo y repartición del trabajo

**FASE 1 MONOHILO:**

**César Fernández Tejero – UO284671**

- Definición de la struct
- Función filter
- Inicialización de punteros de imagen fuente
- Inicialización de punteros de la imagen destino
- Modificación del número de repeticiones para obtener un tiempo significativo
- Documentación mediante comentarios en el código fuente
- Comprobación de la existencia de la imagen fuente

**FASE 2 SIMD:**

**Diego Mieres Molinero – UO282386**

- Función filter y operaciones vectoriales
- Aplicación de las instrucciones simd
- Inicialización de punteros de imagen fuente
- Inicialización de punteros de la imagen destino
- Documentación mediante comentarios en el código fuente

**César Fernández Tejero – UO284671**

- Ajuste del número de repeticiones acorde a la versión monohilo
- Comprobación de la existencia de la imagen fuente

**FASE 2 MULTIHILO:**

**Alejandro Medina Fernández – UO288967**

- Función filter que ejecutan los hilos
- Inicialización de punteros de imagen fuente
- Inicialización de punteros de imagen destino
- Creación de la parte encargada de crear los hilos y trabajar con ellos
- Función encargada de juntar los hilos con pthread_join
- Documentación mediante comentarios en el código fuente
- Comprobación de la existencia de la imagen fuente
- Comprobación de las resoluciones y número de componentes de color

**César Fernández Tejero – UO284671**

- Ajuste del número de repeticiones acorde a la versión monohilo

**MEMORIA:**

**César Fernández Tejero – UO284671**

# Conclusiones

A continuación, se muestra una tabla con los tiempos de las mediciones realizadas sobre cada una de las versiones: single-thread, multi-thread y simd.

| | t_1 (s) | t_2 (s) | t_3 (s) | t_4 (s) | t_5 (s) | t_6 (s) | t_7 (s) | t_8 (s) | t_9 (s) | t_10 (s) | Media t_1 a t_10 (s) | Desv. Típ. t_1 a t_10 (s) | Intervalo de Confianza de Inf. (s) | Sup. (s) | Productividad (tareas/min) | Aceleración (SUT/ref) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SingleThread | 6,7142 | 7,3752 | 6,9199 | 6,7712 | 6,8882 | 6,7399 | 7,3783 | 6,9810 | 6,9518 | 6,6222 | 6,9342 | 0,2596 | 6,6118 | 7,2565 | 8,6528 | - |
| MultiThread | 0,0375 | 0,0379 | 0,0430 | 0,0350 | 0,0393 | 0,0381 | 0,0427 | 0,0452 | 0,0397 | 0,0423 | 0,0401 | 0,0031 | 0,0362 | 0,0439 | 1497,2750 | 173,0394 |
| SIMD | 1,1301 | 1,1750 | 0,3757 | 1,2421 | 1,3488 | 1,1625 | 1,2613 | 1,1477 | 1,3133 | 1,4505 | 1,1607 | 0,2938 | 0,7959 | 1,5255 | 51,6931 | 5,9741 |

Como conclusión podemos observar que distribuir la carga de trabajo entre varios hilos y no recayendo totalmente en uno único, podemos realizar un trabajo mucho mas eficiente, obteniendo unos tiempos de respuesta mucho menores, como podemos observar en los datos recogidos en la tabla anterior. Cabe destacar que la versión multihilo es 173 veces más rápida que la versión monohilo.



Alejando Medina Fernández – UO288967
César Fernández Tejero – UO284671
Diego Mieres Molinero – UO282386
Gustavo Sobrado Aller – UO286277