Trabajo en grupo Arquitectura de Computadores 2021-2022

|  |  |  |
| --- | --- | --- |
| **A** | **Fernández Méndez**  **González Sierra**  **Fuertes Camporro**  **Baños Gómez** | **Lara**  **Fernando José**  **Luis**  **Álvaro** |
| Grupo | Apellidos | Nombre |

|  |  |  |
| --- | --- | --- |
| **71.711.034-R**  **71.754.511-P**  **71.672.531-T**  **71.908.915-J** | **UO276026**  **UO277938**  **UO264691**  **UO245852** | **UO276026@uniovi.es**  **UO277938@uniovi.es**  **UO264691@uniovi.es**  **UO245852@uniovi.es** |
| DNI | Matrícula | e-mail de contacto (@uniovi.es) |

|  |  |  |
| --- | --- | --- |
| **1** | Comparativa de arquitecturas |  |
| Nº Trabajo | Trabajo en grupo de Arquitectura de Computadores | Calificación |

|  |
| --- |
| Comentarios sobre la corrección |
|  |

Asignatura

**ARQUITECTURA DE COMPUTADORES**

**Curso 2021-2022**

|  |  |  |
| --- | --- | --- |
|  | | **Área de Arquitectura y Tecnología de Computadores**  *Departamento de Informática de la Universidad de Oviedo* |
| **1** | | Comparativa de arquitecturas | | |
| Nº Trabajo | | Trabajo en grupo de Arquitectura de Computadores | | |

|  |  |
| --- | --- |
| Apellidos | Nombre |
| **Fernández Méndez** | **Lara** |
| Tareas realizadas | |
| Código del programa | |
| Memoria del trabajo | |
| Mediciones de tiempos | |
|  | |
|  | |
|  | |
|  | |

|  |  |
| --- | --- |
| Apellidos | Nombre |
| **González Sierra** | **Fernando José** |
| Tareas realizadas | |
| Código del programa | |
| Memoria del trabajo | |
| Mediciones de tiempos | |
|  | |
|  | |
|  | |
|  | |
|  | |

|  |  |
| --- | --- |
| Apellidos | Nombre |
| **Fuertes Camporro** | **Luis** |
| Tareas realizadas | |
| Código del programa | |
| Memoria del trabajo | |
| Mediciones de tiempos | |
|  | |
|  | |
|  | |
|  | |
|  | |

|  |  |
| --- | --- |
| Apellidos | Nombre |
| **Baños Gómez** | **Álvaro** |
| Tareas realizadas | |
| Código del programa | |
| Memoria del trabajo | |
| Mediciones de tiempos | |
|  | |
|  | |

**Índice:**

[*0.* *Resumen* 5](#_Toc89084552)

[*1.* *Introducción* 5](#_Toc89084553)

[*2.* *Método de solapamiento: Fusión de capas* 5](#_Toc89084554)

[*3.* *Entorno de trabajo* 6](#_Toc89084555)

[*4.* *Programa, Compilación y Ejecución* 7](#_Toc89084556)

[4.1. Versión Single-Thread 7](#_Toc89084557)

[4.2. Versión con SIMD 8](#_Toc89084558)

[4.3. Versión con multi-thread 8](#_Toc89084559)

[*5.* *Medición de tiempos* 8](#_Toc89084560)

[5.1. SINGLE-THREAD 9](#_Toc89084561)

[5.2. SIMD 9](#_Toc89084562)

[5.3. Multi-Thread 9](#_Toc89084563)

[7. Referencias 10](#_Toc89084564)

# Resumen

El objetivo de este trabajo es realizar un programa de un solo hilo que implemente operaciones sobre componentes de imágenes digitales, en nuestro caso, fusionar dos imágenes en una tercera. Para ello, tenemos un programa base que modificaremos, añadiendo las variables necesarias y cambiando la operación principal. A continuación, ejecutamos el programa y analizamos el tiempo que tarda en ejecutarse (cerrando el resto de programas para que las mediciones no se vean alteradas). También calculamos la media, la desviación típica y el Intervalo de confianza de la media (95%) de el tiempo.

# Introducción

En este trabajo vamos a desarrollar un algoritmo para el procesamiento de imágenes.

En nuestro caso realizaremos una fusión entre dos imágenes.

Posteriormente se realizarán unos análisis de los resultados obtenidos.

# Método de solapamiento: Fusión de capas

El objetivo del algoritmo es realizar una fusión entre dos imágenes de forma solapada, la primera será la escultura “La bailarina” y la segunda un fondo degradado de blanco y negro.

Importante que ambas imágenes tengan el mismo tamaño para aplicar el algoritmo.

La forma de funcionamiento de nuestro algoritmo se basa en recorrer las imágenes simultáneamente, obtener un píxel de cada una y aplicar la fórmula para cada uno de los componentes (R, G, B) de ambas fotos:

Imagen que contiene Texto

Descripción generada automáticamente

X  Imagen 1 Y  Imagen 2 I  Imagen destino

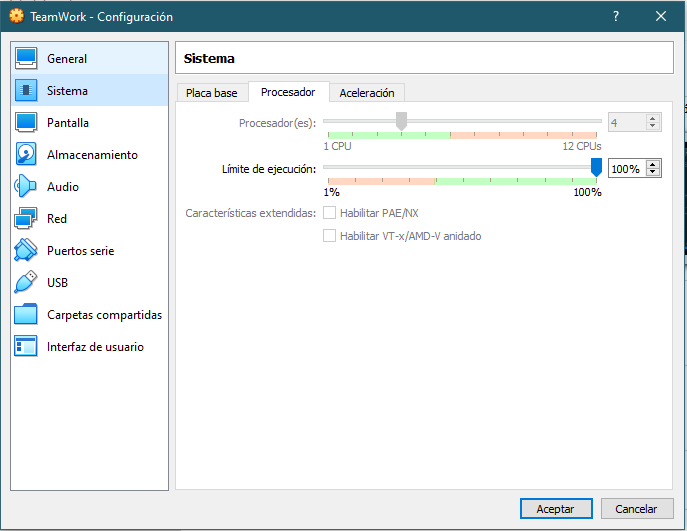
Al terminar de aplicar la formula se habrán fusionado todos los componentes de todos los pixeles de las imágenes, dando lugar a una sola imagen que será el resultado de la fusión solapada de ambas.

# Entorno de trabajo

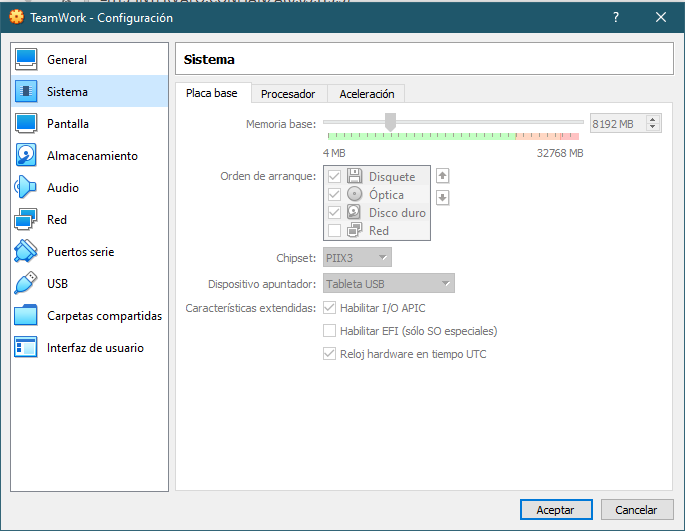
Para la realización de este trabajo hemos utilizado una máquina virtual con el sistema operativo Ubuntu, una distribución de Linux. Tanto Ubuntu como Linux son software libre, es decir, accesible para todos los usuarios de forma gratuita y, además, modificable.

En nuestro caso hemos usado el programa VirtualBox, de Oracle para poder acceder a la máquina virtual.

Las especificaciones del entorno de trabajo son:



**Figura 1.** Número de núcleos de la máquina virtual.



**Figura 2.** Memoria usada para la máquina virtual.

# Programa, Compilación y Ejecución

## Versión Single-Thread

Realizamos un programa en Visual Studio con C++ en el que hacemos la selección de los pixeles de las fotos y su descomposición en componentes mediante punteros. Y aplicando el algoritmo ya mencionado en el apartado 2.0 para cada píxel.

Para la compilación de ese programa utilizamos la ventana de comandos de Ubuntu, el archivo Makefile nos ayuda para este proceso, para ello usamos el comando: “make release”, para obtener el archivo compilado en la carpeta release, el nombre del archivo compilado será “single-thread”, “simd” o “multithread” (Dependiendo de que arquitectura sea).

Tras la compilación del programa podemos ejecutarlo con el comando: “release/***single\_thread***” (la parte en negrita cambia dependiendo de la arquitectura que se quiera ejecutar).

La ejecución nos mostrará por ventana todas las imágenes implicadas en el proceso, las dos de fusión y, tras la aplicación del algoritmo, la resultante, esta última se guardará en la carpeta raíz.

## Versión con SIMD

Para esta versión utilizamos unos juegos de instrucciones denominados SIMD (instrucciones multimedia). Se utilizan sobre operaciones que se repiten ya que permiten realizar estas operaciones sobre un número fijo de elementos a la vez. Se puede considerar una técnica de paralelismo, pero a nivel de datos. Para emplear estas funciones usaremos lo que se denominan [funciones intrínsecas](https://www.redusers.com/noticias/tecnico-pc-conjunto-de-instrucciones-simd/).

En nuestro trabajo se nos ha asignado funciones SIMD que operan con paquetes de datos de 256 bits y usamos el tipo Double que ocupa 64 bits en C. Por lo que cada instrucción SIMD trabajara sobre 4 elementos de tipo Double.

Para desarrollar el diseño de esta solución descompondremos la solución general de transformación de imágenes en operaciones simples usando las instrucciones SIMD.

## Versión con multi-thread

A la hora de realizar un programa para un uso real y profesional debemos tener muy en cuenta el rendimiento, por eso mismo se incorporan los hilos a la hora de ejecutar programas y así acelerar los tiempos.

En nuestro programa, utilizamos una librería pthread.h que nos facilitará la creación de hilos y asignarle una tarea. Esta operación se realiza con el método pthread\_create(…). Introduciendo esta función en un bucle que itere el numero de hilos que queremos procesar podemos hacer que el programa, en nuestro caso de solapamiento con fusión de capas, divida en partes iguales el trabajo a realizar y se lo asigna a cada hilo para que lo ejecute, así consiguiendo que todos los hilos trabajen a la vez en partes distintas del trabajo.

Así en nuestro caso, fusionar dos imágenes, dividimos los pixeles de las fotos por el numero de hilos que vamos a ejecutar para saber el tamaño que tiene que ejecutar cada hilo. Luego en la llamada de cada hilo se le pasa un objeto ImgInfo que tiene la información necesaria para ejecutar el algoritmo, es decir, donde empieza, donde acaba y las direcciones de los colores donde tiene que escribir. Para finalizar, esperamos que terminen todos los hilos de ejecutar y con pthread\_join(...) pasándole cada hilo los terminamos.

# Medición de tiempos

Tomamos tiempos diez veces para realizar una media con la que trabajar.

De ella a su vez podemos obtener una desviación típica, es decir lo que varía los datos respecto de la media.

Luego calculamos los intervalos de confianza con un nivel de confianza del 95%.

## Single-Thread



**Figura 3.** Tiempos medidos, mono hilo.

Tabla

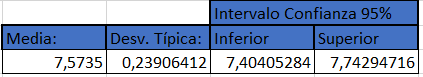
Descripción generada automáticamente

**Figura 4.** Media, Desviación Típica e Intervalo de Confianza 95% inferior y superior. (Mono hilo).

## SIMD



**Figura 5.** Tiempos medidos, SIMD.

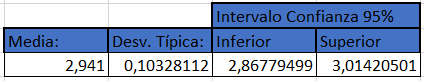


**Figura 6.** Media, Desviación Típica e Intervalo de Confianza 95% inferior y superior. (SIMD).

## Multi-Thread



**Figura 7.** Tiempos medidos, multihilo.



**Figura 8.** Media, Desviación Típica e Intervalo de Confianza 95% inferior y superior. (Multihilo).

1. Conclusiones

La realización de los filtros en imágenes es un proceso muy sencillo ya que solo utilizamos un código relativamente básico, con un algoritmo conocido. Por lo que modificando únicamente el algoritmo podríamos conseguir otros filtros.

# Referencias

Ubuntu: <https://ubuntu.com>

Linux: <https://www.linux.org>

Oracle VirtualBox: <https://www.virtualbox.org>

Funciones intrínsecas: <https://www.redusers.com/noticias/tecnico-pc-conjunto-de-instrucciones-simd/>