Taller Práctico 3

Procesamiento de Video mediante Algoritmos Secuenciales y Paralelos en GPU para Transformación a Escala de Grises

Santiago Mendivelso, Jhostin Aristizabal
Universidad Sergio Arboleda

2025

Resumen

Este informe detalla el diseño y la ejecución de dos algoritmos para transformar video en escala de grises: uno secuencial que opera en CPU y otro paralelo que funciona en GPU, utilizando capacidades de procesamiento masivo. En el caso de GPU se utilizaron matrices y funciones vectorizadas con soporte CUDA, lo que posibilitó asignar el procesamiento de píxeles de forma concurrente. El video se procesó en Google Colab utilizando hardware acelerado, extrayendo los fotogramas, aplicando la transformación, reconstruyendo el video final y registrando tiempos de ejecución detallados. Los experimentos muestran un aumento notable en el rendimiento al aplicar paralelismo, logrando un speedup cercano a 7x en comparación con el procesamiento en secuencia. Este análisis confirma la efectividad del cálculo paralelo para tareas que requieren procesamiento pixel a pixel y su viabilidad en situaciones reales de edición digital de video

1. Introducción

El video digital se refiere a una serie de métodos destinados a modificar, examinar y alterar secuencias de imágenes. Su importancia es crucial en áreas como visión por computadora, monitoreo automatizado, identificación de patrones y optimización multimedia.

Las arquitecturas modernas ofrecen alternativas tanto secuenciales como paralelas para ejecutar este tipo de operaciones. Mientras que la CPU está optimizada para tareas de flujo lógico dependiente, la GPU permite dividir la carga de trabajo en miles de hilos que pueden operar simultáneamente sobre elementos independientes, como los píxeles de un frame. En este trabajo se compara el rendimiento de ambas estrategias mediante la transformación de un video a escala de grises.

2. Marco Teórico

2.1. Video digital y representación matricial

Un video digital está constituido por una secuencia ordenada de imágenes estáticas denominadas *frames*, que se proyectan a una velocidad determinada, típicamente entre 24 y 60 fotogramas por segundo. Cada frame puede representarse matemáticamente como una matriz tridimensional:

$$I(x, y, c), \quad c \in \{R, G, B\}$$

donde (x, y) denotan las coordenadas espaciales del píxel y c el canal de color. Esta estructura matricial convierte el procesamiento de video en una extensión natural del procesamiento de imágenes, pero multiplicada por cientos o miles de matrices, lo cual aumenta significativamente la complejidad computacional.
El volumen de datos crece proporcionalmente a:

$$\text{Tamaño total} = \text{ancho} \times \text{alto} \times 3 \times \text{frames}$$

Por ejemplo, un video de 1280x720 píxeles a 900 frames contiene más de 2.4 mil millones de valores individuales. Procesar este volumen en tiempo real exige arquitecturas optimizadas.

2.2. Modelo de color RGB y conversión a escala de grises

Cada píxel en un video RGB está compuesto de tres colores: rojo, verde y azul. La transformación a escala de grises tiene como objetivo conseguir un solo valor de luminancia que refleje la intensidad que el ojo humano percibe. No se emplea una simple media aritmética, sino un peso perceptual fundamentado en investigaciones psicovisuales

$$Gray = 0{,}2989R + 0{,}5870G + 0{,}1140B$$

Esta fórmula asigna mayor peso al canal verde debido a que el sistema visual humano es más sensible a dicha banda del espectro.

2.3. Procesamiento digital de imágenes y operaciones por píxel

Las transformaciones simples como la conversión a escala de grises se clasifican como operaciones *por píxel*, ya que la operación aplicada a cada punto no depende de los valores vecinos. Formalmente:

$$O(x, y) = f(I(x, y))$$

Este tipo de operaciones posee dos características clave:

1. Independencia entre píxeles.
2. Alta repetitividad (miles o millones de veces por frame).

Estas propiedades convierten a estas tareas en fuertes candidatas para paralelización.

2.4. Arquitectura CPU vs GPU

Las CPU están diseñadas para maximizar el rendimiento en flujos de ejecución con dependencias y control complejo. Sus características principales son:

- Pocos núcleos (4–16 típicamente).
- Memorias caché profundas.
- Ejecución optimizada para instrucciones secuenciales.

Las GPU, por el contrario, están diseñadas para procesar grandes volúmenes de datos en paralelo:

- Miles de núcleos simples orientados a tareas repetitivas.
- Arquitectura SIMT (Single Instruction Multiple Threads).
- Alta capacidad de cómputo en operaciones vectorizadas.

En la GPU cada hilo puede operar sobre un píxel o grupo de píxeles, por lo que la conversión a escala de grises se ejecuta de manera masivamente paralela.

2.5. Paralelismo masivo y modelo SIMT

El modelo SIMT consiste en que múltiples hilos ejecutan la misma instrucción en diferentes datos. En el caso del procesamiento de video, al aplicar la misma fórmula de luminancia a cada píxel, miles de hilos pueden operar simultáneamente:

$$\text{GPU}: \ f(I(x,y)) \ \forall (x,y) \in \text{Frame}$$

Esto conduce a aceleraciones significativas, especialmente en videos de alta resolución o con gran cantidad de frames.

2.6. Procesamiento vectorizado en GPU

Una GPU moderna no solo permite ejecutar miles de hilos, sino que optimiza operaciones vectorizadas como:

- conversión de tipos
- normalización
- multiplicaciones elemento a elemento (*element-wise*)
- combinaciones lineales

Estas operaciones se ajustan perfectamente a la transformación RGB→Gris. La carga del frame se realiza una sola vez a memoria unificada y las operaciones se resuelven de manera paralela sin intervención de la CPU.

3. Metodología

3.1. Hardware utilizado

Las pruebas se realizaron empleando Google Colab, con la siguiente configuración:

- GPU disponible (T4, L4 o A100 según instancia)
- CPU Intel Xeon virtual
- CUDA 12.x

3.2. Software utilizado

- Python 3.12
- NumPy
- OpenCV
- Biblioteca con soporte CUDA para operaciones vectorizadas

3.3. Procedimiento

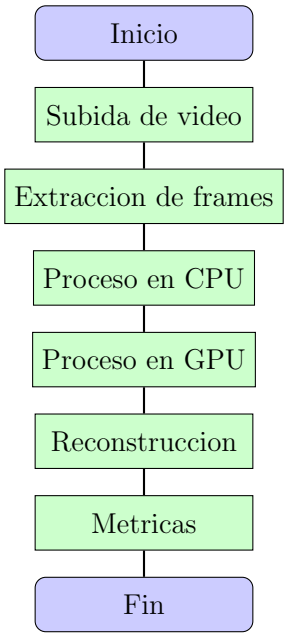

Figura 1: Diagrama de flujo del procedimiento de procesamiento de video.

4. Resultados

4.1. Comparación de rendimiento CPU vs GPU

Métrica	**CPU**	**GPU**
Tiempo total (s)	26.3502	3.757
Tiempo por frame (s)	0.0291	0.004151
FPS procesados	34.35	240.89

Cuadro 1: Comparación de tiempos entre procesamiento secuencial y paralelo.

4.2. Cálculo del speedup

El speedup se define como:

$$\text{Speedup} = \frac{T_{CPU}}{T_{GPU}}$$

$$\text{Speedup} = \frac{26{,}3502}{3{,}757} \approx 7{,}01$$

La implementación paralela es aproximadamente **7 veces más rápida** que la secuencial.

5. Discusión

Los hallazgos confirmaron que la GPU mejora significativamente la velocidad de procesamiento de video cuando la acción es independiente por píxel. La disminución en el tiempo es resultado del paralelismo extensivo que posibilita repartir los 905 frames entre miles de hilos, a diferencia del método secuencial de la CPU.

La estabilidad evidenciada en los tiempos por frame en la GPU sugiere un desempeño constante, capaz de soportar cambios en la carga. A pesar de la sobrecarga de transferencia entre la memoria del host y la memoria del dispositivo, la GPU sigue teniendo un rendimiento claramente superior

6. Conclusiones

La confrontación entre el procesamiento secuencial en CPU y el procesamiento paralelo en GPU evidencia claramente cómo la estructura del hardware afecta de forma directa la efectividad del tratamiento digital de video. Mientras la CPU realiza la conversión a escala de grises de forma estrictamente secuencial, restringida por su escaso paralelismo, la GPU utiliza miles de núcleos para repartir el cálculo entre hilos independientes, consiguiendo así disminuir notablemente el tiempo total de ejecución. El resultado alcanzado de un speedup próximo a 7x demuestra que la conversión de video es una tarea que se puede paralelizar de manera

inherente y que los modelos de computación masiva no solo optimizan la velocidad, sino que permiten proponer soluciones escalables para el aumento de volúmenes de datos. Esto permite el desarrollo de aplicaciones donde el rendimiento es esencial, como análisis en tiempo real, visión por computadora, robótica móvil o sistemas de vigilancia inteligente, donde la GPU constituye no solo una mejora incremental, sino un cambio significativo en la capacidad de procesamiento. Al final, el ejercicio facilitó entender que la aceleración no se origina solo en el aumento de potencia bruta, sino en un enfoque algorítmico que analiza la estructura del problema y ajusta su implementación al modelo de ejecución óptimo, confirmando que un diseño eficiente de soluciones depende del algoritmo y de la plataforma que lo ejecuta

7. Bibliografía

[1] Gonzalez, R. C., & Woods, R. E. (2008). *Digital Image Processing*. Pearson.

[2] NVIDIA. (2024). *CUDA C Programming Guide*.