Informe de Laboratorio

Taller Práctico 3: Procesamiento de Video en Algoritmos Paralelos: Transformando un Video a Escala de Grises

Juan Hurtado, Miguel Flechas, Andres Castro

Universidad
Sergio Arboleda

Noviembre

2025

Computación de Alto Rendimiento (HPC)

1. TÍTULO Y OBJETIVOS

Título

Implementación y Comparación de Algoritmos Secuencial y Paralelo Multicore para el Procesamiento de Video en Escala de Grises

Objetivo General

Implementar y comparar la eficiencia de un algoritmo secuencial y uno paralelo multihilos al procesar un video, transformando el video original a escala de grises, y demostrar el speedup (aceleración) logrado con el paralelismo.

Objetivos Específicos

- Desarrollar un algoritmo secuencial para convertir un video a color en escala de grises
- Implementar un algoritmo paralelo multicore que aproveche múltiples núcleos de CPU
- Medir y comparar los tiempos de ejecución de ambos algoritmos
- Calcular el speedup y la eficiencia del procesamiento paralelo
- Generar un video resultante en escala de grises

2. MARCO TEÓRICO

Procesamiento de Video

El Procesamiento de Video es el conjunto de técnicas y algoritmos utilizados para manipular, analizar y mejorar secuencias de imágenes digitales (video). Constituye un campo fundamental en la ingeniería y la informática, con aplicaciones que van desde la compresión y transmisión de datos hasta el análisis de movimiento y la visión por computadora.

Conceptos Fundamentales

Video Digital: Una secuencia de fotogramas (imágenes fijas) mostrados a una frecuencia determinada (tasa de frames por segundo, FPS). Cada fotograma está compuesto por píxeles que almacenan información de color e intensidad.

Escala de Grises: Transformación que reduce la información de color de una imagen a valores de intensidad luminosa, simplificando el procesamiento y reduciendo la cantidad de datos.

Fases del Procesamiento de Video

Captura/Entrada: Adquisición de la secuencia de video (archivo, cámara, streaming)

Preprocesamiento: Ajustes iniciales como normalización o escalado

Análisis/Manipulación: Aplicación del algoritmo central (conversión a escala de grises)

Codificación/Salida: Formato final para almacenamiento o visualización

Procesamiento Paralelo

El procesamiento paralelo permite dividir una tarea en múltiples subtareas que se ejecutan simultáneamente en diferentes núcleos de CPU, aprovechando el hardware moderno para reducir significativamente los tiempos de ejecución.

Speedup: Métrica que indica cuántas veces más rápido es un algoritmo paralelo comparado con su versión secuencial.
Fórmula: Speedup = Tiempo_Secuencial / Tiempo_Paralelo

Eficiencia: Porcentaje que indica qué tan bien se aprovechan los recursos de hardware.
Fórmula: Eficiencia = (Speedup / Núcleos_CPU) × 100%

3. METODOLOGÍA

3.1 Configuración del Hardware

- Sistema Operativo: Windows/Linux
- Procesador: CPU con 2 núcleos disponibles
- Memoria RAM: Suficiente para procesamiento de video

3.2 Configuración del Software

- Entorno de Desarrollo: Google Colab
- Lenguaje de Programación: Python 3.x
- **Librerías Utilizadas:**

- **OpenCV (cv2):** Procesamiento de imágenes y video
- **NumPy:** Operaciones numéricas y manipulación de arrays
- **Multiprocessing:** Implementación del procesamiento paralelo
- **Time:** Medición de tiempos de ejecución

3.3 Descripción del Video de Entrada

- Archivo: rana.mp4
- Resolución: 576x576 píxeles
- FPS (Frames por segundo): 30.00
- Total de Frames: 140
- Duración: 4.67 segundos

3.4 Algoritmos Desarrollados

Algoritmo 1: Procesamiento Secuencial

El algoritmo secuencial procesa cada frame del video uno por uno de forma iterativa:

1. Extracción de frames: Descomponer el video en imágenes JPG individuales
2. Procesamiento individual: Para cada frame:
   - Leer la imagen en formato BGR
   - Convertir de BGR a RGB
   - Calcular el promedio de los canales RGB
   - Guardar la imagen procesada
3. Reconstrucción: Unificar todas las imágenes en un video MP4

Complejidad: O(n) donde n es el número de frames

Algoritmo 2: Procesamiento Paralelo Multicore

El algoritmo paralelo distribuye el procesamiento de frames entre múltiples núcleos de CPU:

4. División de trabajo: Crear una lista de tareas (frames a procesar)
5. Pool de procesos: Crear un pool con N procesos (N = número de núcleos)
6. Distribución paralela: Asignar frames a diferentes procesos
7. Procesamiento simultáneo: Cada núcleo procesa frames independientemente

8. Sincronización: Recolectar resultados de todos los procesos
9. Reconstrucción: Unificar frames procesados en video final

Ventajas:

- Aprovechamiento de múltiples núcleos de CPU
- Reducción significativa del tiempo de procesamiento
- Escalabilidad según hardware disponible

4. RESULTADOS

4.1 Extracción de Frames

Se extrajeron exitosamente 140 frames del video original, guardados en formato JPG con nomenclatura secuencial (frame_000000000.jpg a frame_000000139.jpg).

4.2 Tiempos de Ejecución

Algoritmo	Tiempo de Ejecución	Frames Procesados
Secuencial	4.4902 segundos	140/140
Paralelo (2 núcleos)	3.3738 segundos	140/140

Tiempo promedio por frame:

- Secuencial: 0.0321 segundos/frame
- Paralelo: 0.0241 segundos/frame

4.3 Métricas de Rendimiento

Speedup Obtenido: 1.33x

Eficiencia: 66.54%

Tiempo Ahorrado: 1.1164 segundos

4.4 Video Resultante

Se generó exitosamente el video video_escala_grises.mp4 con las siguientes características:

- Resolución: 576x576 píxeles (igual al original)
- FPS: 30 (igual al original)
- Total de frames: 140
- Formato: Escala de grises (monocromático)

5. ANÁLISIS DE RENDIMIENTO

5.1 Comparación de Algoritmos

El algoritmo paralelo demostró ser 1.33 veces más rápido que el algoritmo secuencial, lo que representa una mejora del 33% en el tiempo de procesamiento.

Interpretación del Speedup:

- Un speedup de 1.33x indica que el procesamiento paralelo reduce el tiempo en aproximadamente 25% (de 4.49s a 3.37s)
- Aunque el speedup ideal con 2 núcleos sería 2.0x, un valor de 1.33x es razonable considerando:
  - Overhead de creación y sincronización de procesos
  - Operaciones de I/O (lectura/escritura de archivos)
  - Limitaciones de memoria y ancho de banda

5.2 Eficiencia del Paralelismo

La eficiencia de 66.54% indica que cada núcleo está siendo aprovechado al 66.54% de su capacidad máxima teórica.

Análisis:

- Buena paralelización (>50%)
- El overhead del procesamiento paralelo consume aproximadamente 33% del beneficio potencial
- Factores que afectan la eficiencia:
  - Creación y destrucción de procesos
  - Comunicación entre procesos

- Sincronización de resultados
- Operaciones de I/O no paralelizables

5.3 Ventajas y Limitaciones

Ventajas del Algoritmo Paralelo:

- Reducción significativa del tiempo de procesamiento
- Aprovechamiento eficiente del hardware moderno
- Escalable a videos más largos y resoluciones más altas
- Mismo resultado final que el algoritmo secuencial

Limitaciones Identificadas:

- Overhead de creación de procesos (más notable en videos cortos)
- Mayor consumo de memoria (múltiples procesos simultáneos)
- Complejidad de código aumentada
- Operaciones de I/O siguen siendo secuenciales

6. CONCLUSIONES

Conclusiones Generales

1. Implementación Exitosa: Se logró implementar exitosamente tanto el algoritmo secuencial como el paralelo para la conversión de video a escala de grises, cumpliendo con los objetivos planteados.

2. Beneficio del Paralelismo: El algoritmo paralelo demostró ser 1.33x más rápido que el secuencial, comprobando que el procesamiento paralelo ofrece ventajas reales en el procesamiento de video.

3. Eficiencia Aceptable: Con una eficiencia del 66.54%, se considera que el algoritmo paralelo aprovecha adecuadamente los recursos de hardware disponibles.

4. Calidad Preservada: Ambos algoritmos produjeron resultados idénticos en términos de calidad visual del video resultante, validando la correctitud de la implementación paralela.

Aprendizajes sobre Procesamiento Paralelo

1. Overhead vs. Beneficio: El procesamiento paralelo tiene un costo de overhead que debe ser considerado, especialmente para tareas pequeñas.

2. Granularidad de Tareas: La división de trabajo en el nivel de frames es efectiva, ya que cada frame puede procesarse independientemente.

3. Limitaciones de I/O: Las operaciones de entrada/salida representan un cuello de botella que limita el speedup máximo alcanzable.

4. Escalabilidad del Hardware: El rendimiento del algoritmo paralelo es directamente proporcional al número de núcleos disponibles.

Reflexión Final

Este laboratorio demostró de manera práctica los beneficios del procesamiento paralelo en aplicaciones del mundo real. Aunque el speedup obtenido (1.33x) puede parecer modesto, representa una mejora significativa en términos de tiempo de procesamiento. En aplicaciones a gran escala o con hardware más potente, estos beneficios se magnifican considerablemente.

El dominio de técnicas de programación paralela es fundamental en la era del big data y la computación de alto rendimiento, donde la capacidad de procesar grandes volúmenes de datos de manera eficiente es crucial.