# Informe Parcial 1 — Adaptación basada en el repositorio de J. M. López

**Asignatura:** Programación Paralela

**Autor:** Jose Miguel López

**Fecha:** 31 de agosto de 2025

**Repositorio:** <https://github.com/jmlopezrianiJave/Midterm-1>

**Video:** https://youtu.be/YxJ0mutCZsk

## Resumen

Este documento el código fuente y resultados experimentales del parcial. Se aplican filtros convolucionales a imágenes secuencialmente, y con paralelización usando pthreads y OpenMP. Los resultados favorecen la paralelización para este tipo de tareas, con OpenMP probando ser el más efectivo en el speedup visto.

## Introducción

El objetivo del trabajo es implementar y comparar varios métodos de paralelismo al filtrado de imágenes en formatos PGM y PPM. Se busca medir el impacto en tiempos de ejecución y responder la pregunta de interés: ¿Cómo influye la programación paralela, utilizando OpenMP ó Pthreads y MPI ( En un entorno simulado con docker ), en el tiempo de ejecución del filtrado de imágenes PPM y PGM en comparación con una implementación secuencial?

## Metodología experimental

* Se utilizaron varias imágenes de prueba (Lena, Fruit, PUJ) en formatos PGM y PPM.
* Para cada implementación y filtro se midieron tiempos de ejecución usando relojes de alta resolución de la librería chrono.
* Las mediciones se realizaron repetidas veces y se reporta el valor promedio representativo.

## Resultados experimentales

### Tiempos — Implementación Secuencial (ms)

* **Lena PGM**: Blur 128 ms | Laplace 120 ms | Sharpen 132 ms
* **Lena PPM**: Blur 23 ms | Laplace 23 ms | Sharpen 26 ms
* **Fruit PGM**: Blur 210 ms | Laplace 179 ms | Sharpen 202 ms
* **Fruit PPM**: Blur 559 ms | Laplace 484 ms | Sharpen 561 ms
* **PUJ PGM**: Blur 553 ms | Laplace 484 ms | Sharpen 525 ms
* **PUJ PPM**: Blur 1498 ms | Laplace 1428 ms | Sharpen 1476 ms

### Tiempos — Pthreads (ms)

* **Lena PGM**: Blur 103 ms | Laplace 102 ms | Sharpen 121 ms
* **Lena PPM**: Blur 26 ms | Laplace 20 ms | Sharpen 30 ms
* **Fruit PGM**: Blur 175 ms | Laplace 162 ms | Sharpen 173 ms
* **Fruit PPM**: Blur 516 ms | Laplace 440 ms | Sharpen 486 ms
* **PUJ PGM**: Blur 528 ms | Laplace 427 ms | Sharpen 469 ms
* **PUJ PPM**: Blur 1367 ms | Laplace 1244 ms | Sharpen 1430 ms

### Tiempos — OpenMP (ms) (Tres filtros aplicados simultáneamente)

* **Lena PGM**: 100 ms
* **Lena PPM**: 25 ms
* **Fruit PGM**: 149 ms
* **Fruit PPM**: 439 ms
* **PUJ PGM**: 402 ms
* **PUJ PPM**: 1180 ms

## Análisis

* Las versiones paralelas (Pthreads y OpenMP) muestran una reducción consistente en los tiempos respecto a la versión secuencial en la mayoría de casos.
* Open MP muestra la mayor mejoría, ya que mostró tiempos de ejecución menores a los de Pthreads y la versión secuencial. Además, esto es con las versiones de Pthreads y secuencial aplicando un filtro a la vez, mientras OpenMP estaba aplicando los tres filtros a la vez.
* La mayor mejoría se vio en la aplicación de filtros de puj, que era la imagen más grande.
* A esta escala, es más beneficioso paralelizar la aplicación de múltiples filtros, en lugar de usar la paralelización para dividir el trabajo que toma filtrar una sola foto.
* En fotos de mucha mayor escala podría probar ser más efectivo el tipo de paralelización usada en el módulo de pthreads contra la usada en OpenMP.