**AC Teamwork – Group A PL1: Team Contributions, Filter Explanation, and Performance Analysis**

**Blend Filter Explanation**

The project implements a **blend filter** that combines two images, producing a visually fused output. The blending algorithm computes pixel values of the destination image as a weighted combination of corresponding pixel values from two source images (X and Y). The formula used in this filter is:

This computation is applied individually to the **Red**, **Green**, and **Blue** components of each pixel. The algorithm ensures seamless blending by preserving pixel intensity within the valid range of [0, 255].

The program processes each pixel sequentially or in parallel, depending on the implementation (single-threaded, SIMD, or multithreaded), and writes the processed pixels to the destination image.

**Patrón de fondo

Descripción generada automáticamenteImagen que contiene exterior, edificio, camino, calle

Descripción generada automáticamenteInputs:**

Una torre de un edificio

Descripción generada automáticamente con confianza media**Output: Same for the 3 programs**

**Teamwork Distribution**

Our team strategically divided the workload to maximize efficiency and leverage individual strengths:

1. **Single-Threaded Implementation**:
   * Served as the baseline implementation for performance comparisons.
   * Responsible member: **Álvaro Puebla Ruisánchez**
2. **SIMD Implementation**:
   * Utilized vectorized instructions to process multiple pixels simultaneously.
   * Responsible members: **Sofía Leiras Núñez** and **Ignacio Hoyos Diego**.
3. **Multithreaded Implementation**:
   * Distributed the workload across multiple CPU cores to maximize concurrency.
   * Responsible member: **Daniel Álvarez Menéndez**.

**Contribution Table**

|  |  |  |
| --- | --- | --- |
| **Team Member** | **Task** | **Contribution (%)** |
| Álvaro Puebla | Single-threaded Program | 25% |
| Sofía Leiras | SIMD Implementation | 25% |
| Ignacio Hoyos | SIMD Implementation | 25% |
| Daniel Álvarez | Multithreaded Program | 25% |

**Performance Results and Analysis**

The program's performance was measured across 10 repetitions for each implementation. The results, including the average execution times, standard deviations, and 95% confidence intervals, are shown below:

**Performance Data Table**

Texto

Descripción generada automáticamenteTexto

Descripción generada automáticamente

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente**Texto

Descripción generada automáticamente

**Insights**

* **Single-Threaded Implementation**:
  + Provides a sequential baseline for comparison.
  + Took the longest execution time, as expected.
* **SIMD Implementation**:
  + Achieved a speedup of approximately 2.42× compared to the single-threaded version.
  + Demonstrated the advantages of vectorized processing.
* **Multithreaded Implementation**:
  + Achieved an extraordinary speedup of approximately 50.72×.
  + Fully utilized multi-core capabilities to process pixels in parallel.

**Comparation with** [**https://www.imgonline.com.ua/eng/similarity-percent.php**](https://www.imgonline.com.ua/eng/similarity-percent.php) **:**

\*The diffImages didn’t work so we found this web

* **Single-Threaded vs SIMD**:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Imagen que contiene Diagrama

Descripción generada automáticamente

* **Single-Threaded vs Multithreaded:**

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Imagen que contiene Diagrama

Descripción generada automáticamente

The 3 images are the same, so the programs give the same output given the same inputs.

**Conclusion**

The blend filter was successfully implemented in three different modes, each showcasing varying levels of performance improvement. The multithreaded implementation significantly outperformed the other methods, proving the power of parallelization. The SIMD implementation also showed notable improvements by leveraging data-level parallelism.

This analysis highlights the impact of optimizing computational algorithms using advanced programming techniques, paving the way for future enhancements in performance-critical applications.