IBARRA - ECUADOR
Facultad de Ingeniería en Ciencias Aplicadas
Innovation & Development in Enginnering in Applied Sciences
IDEAS
VOL. 4 Nº2 - 2022

Segmentación de imágenes agrícolas adquiridas con drone mediante algoritmos paralelos

http://doi.org/10.53358/ideas.v4i2.861

Patricia Ponce[1], MacArthur Ortega-Bustamante[1], Marco Pusdá-Chulde[1]
[1]Facultad de Ingeniería en Ciencias Aplicadas, Universidad Técnica del Norte, Ibarra, Ecuador
[1]*{epponceg, mc.ortega, mrpusda}@utn.edu.ec*

Fecha de envío, marzo 21, 2023 - Fecha de aceptación, abril 12, 2023 - Fecha de publicación, Abril 21, 2023

RESUMEN

Las imágenes obtenidas con drones desde perspectiva vertical presentan información importante de los cultivos agrícolas, lo que permite sistematizar diversas actividades relacionadas con la agricultura de precisión (AP). Los algoritmos secuenciales desarrollados en lenguajes de programación tradicionales consumen tiempos y recursos de hardware, altos en el procesamiento de imágenes digitales de grandes tamaños. Para reducir los tiempos de cómputo, se desarrolló un algoritmo paralelo, que permite segmentar imágenes utilizando la librería OpenMP. OpenMP es una librería compatible con lenguajes de programación de bajo nivel, que permiten la implementación de algoritmos paralelos en lenguajes C o C++ en entornos Linux para arquitecturas multicore. Para implementar el algoritmo secuencial mediante paralelismo, fue necesario dividir en varias tareas más pequeñas y ejecutarlas en varios núcleos disponibles en procesadores multicore, a fin de mejorar la velocidad de procesamiento. Las métricas utilizadas (tiempo de ejecución, aceleración, eficiencia y costo computacional) permitieron evaluar el rendimiento de los algoritmos con imágenes de diferentes dimensiones, obteniendo resultados favorables que verifica la mejora del algoritmo paralelo. La paralelización de algoritmos secuenciales muestra una importante reducción de los tiempos de ejecución (66.37%) con imágenes grandes y (74.73%) con imágenes pequeñas utilizando el número máximo de núcleos (8).

Palabras Clave: OpenMP, segmentación de imágenes, drone, agricultura de precisión, paralelismo.

Abstract: The images obtained with drones from vertical perspective present important information of agricultural crops, allowing the systematization of several activities related to precision agriculture (PA). Sequential algorithms developed in traditional programming languages consume elevated time and hardware resources in the processing of large digital images. To reduce computation time, a parallel algorithm was developed, which allows image segmentation using the OpenMP library. OpenMP is a library compatible with low-level programming languages, which allows the implementation of parallel algorithms in C or C++ languages in Linux environments for multicore architectures. To implement the sequential algorithm using parallelism, it was necessary to split into several smaller tasks and execute them on several cores available on multicore processors, in order to improve the processing speed. The metrics used (execution time, acceleration, efficiency and computational cost), allowed evaluating the performance of the algorithms with images of different dimensions, obtaining favorable results that verify the improvement of the parallel algorithm. The parallelization of sequential algorithms shows a significant reduction of execution times (66.37%) with large images and (74.73%) with small images using the maximum number of cores (8).

Keywords: OpenMP, image segmentation, drone, precision agriculture, parallelism.

Autor de correspondencia:
Marco Pusdá, mrpusda@utn.edu.ec

Introducción

La segmentación de imágenes digitales son herramientas de procesamiento que permiten dividir en diferentes secciones las imágenes originales. Los resultados de la segmentación de imágenes se pueden utilizar en la sistematización de tareas agrícolas. El procesamiento de imágenes agrícolas obtenidas mediante recursos tecnológicos no tripulados (drones) proporcionan información importante que puede utilizarse para varios propósitos en la toma de decisiones más adecuadas y eficientes por parte de los agricultores [1], [2]. Debido a que las imágenes obtenidas mediante drones son de alta calidad y dimensiones grandes, requiere de gran capacidad de procesamiento, esto conduce a la necesidad de ampliar tiempos de ejecución y alto consumo de recursos computacionales. La computación de alto rendimiento (HPC- High Performance Computing) puede acelerar drásticamente el procesamiento de imágenes disminuyendo considerablemente los tiempos de ejecución [3].

Las computadoras personales actuales utilizan procesadores con múltiples núcleos en un solo procesador, como principio fundamental de computación de alto rendimiento, este tipo de arquitecturas facilitan la sistematización de tareas que requieren alto esfuerzo computacional [4]. La utilización de HPC permite diseñar algoritmos de visión por computador, usando paralelismo sobre plataformas multiprocesador (multicore), lo que permite mejorar el rendimiento en el procesamiento de imágenes digitales en diferentes tareas agrícolas [5]–[7]. Las imágenes incluyen una gran cantidad de información y, por lo tanto, la mayoría de las técnicas de procesamiento y análisis de imágenes no son rápidas; la aceleración de estas técnicas es de gran importancia y vital para diferentes aplicaciones informáticas.

OpenMP es una solución eficiente para programación paralela en sistemas de memoria compartida basado en librerías y directivas de lenguajes de bajo nivel (C, C++, Fortran), proporcionando flexibilidad para transformar algoritmos seriales en algoritmos paralelos en infraestructuras multicore [8], [9]. El paralelismo de OpenMP se realiza mediante hilos que permiten gestionar de manera independiente el trabajo total del algoritmo en cada núcleo

del CPU ejecutando un subproceso diferente. Los modelos de programación utilizados por lenguajes de memoria compartida como OpenMP proporcionan mecanismos eficientes para aprovechar las capacidades de los núcleos, esto permite potenciar el rendimiento de las aplicaciones para la solución de problemas computacionales [8], [10], [11].

La AP se considera como un sistema agrícola alternativo sostenible que utiliza diferentes métodos o herramientas tecnológicas para mejorar y optimizar el rendimiento de los cultivos, disminuir costos de producción y reducir los impactos ambientales. La alimentación mundial requiere que, la producción agrícola minimice los impactos ecológicos negativos de sus actividades y sea competitiva en mercados globalizados cada vez más exigentes en precios y calidades [12]–[14]. Aprovechando los avances tecnológicos, drones, arquitecturas heterogéneas (Multicore, GPU, FGPA) y técnicas de inteligencia artificial (visión computarizada, machine learning, deep learning), ha permitido sistematizar una variedad de actividades agrícolas, tales como detección de enfermedades, recuento de plantas e identificación de malezas, plagas de insectos aplicados en diferentes cultivos [10], [15]–[19].

En los últimos tiempos la AP se acoge como una moderna herramienta agrícola, la cual inicia con la mecanización y sistematización de la producción, enfocándose en la nutrición del suelo, la fertilización del cultivo, la productividad, las ventajas económicas y la minimización del impacto ambiental, producto de las labores de campo, como también, la gestión y manejo de semillas, la distribución de plantación, control de siembra y administración de los recursos utilizados en los cultivos [20]–[23]. Los sistemas de información geográfica permiten participar en distintas etapas del proceso productivo de la agricultura de precisión por medio de imágenes satelitales, mapas de prescripción de siembra variables, procesamiento de monitores de rendimiento, entre otras actividades. El uso de las imágenes satelitales juntamente con los sistemas de información geográfica (SIG), los sistemas de posicionamiento global (GPS), la teledetección y la programación, ofrecen servicios de consultoría, desarrollo de sistemas, tecnología geoespacial aplicada, sistemas de análisis de datos, visitas y soluciones a campo, capacitación y soporte [24]–[26].

La segmentación de imágenes digitales agrícolas ha sido investigada y aplicada en varios trabajos de investigación, mediante métodos tradicionales y también métodos innovadores con técnicas de inteligencia artificial [27]. En el trabajo [28] se realizó un método utilizando el algoritmo K-Means para segmentar imágenes, permitiendo la aceleración mediante paralelización efectiva en procesadores Xeon Phi y GPU. En [29] se realizó un modelo para clasificar sistemas de riego mediante segmentación de imágenes y aprendizaje profundo aplicable a diferentes tipos de cultivos agrícolas. En [29] se utilizó la segmentación de imágenes satelitales para delimitar campos de cultivo mediante mapeos de áreas extensas de cultivos, logrando determinar los recursos agrícolas necesarios (agua, nutrientes y otras características) para mejorar la producción agrícola.

En el presente artículo, se desarrolla un algoritmo secuencial y otro paralelo para procesar imágenes digitales agrícolas de tres dimensiones en pixeles (1920 x 1080, 1280 x 720, 640x 360). Los algoritmos se implementaron en OpenMP y se evaluaron en varios núcleos disponibles en procesadores multicore. Las métricas de rendimiento utilizadas muestran un mejor rendimiento en tiempos de ejecución y aceleración a medida que se utiliza mayor cantidad núcleos en el algoritmo paralelo con respecto al secuencial.

Materiales y métodos

La segmentación de imágenes es el proceso de fraccionar una imagen digital en sus objetos constituyentes para identificar los principales elementos que componen la escena, es el primer paso hacia una detección efectiva de los objetos presentes en la imagen. Cuanto más precisa sea la fase de segmentación, más probable es que el proceso de reconocimiento de imágenes tenga éxito. Se consideró la segmentación simple basada en el color, dividiendo los elementos de interés y discriminantes en regiones blancas y negras respectivamente, para convertir los datos en un entorno lógico de ceros y unos, y, permitir al sistema reconocer y extraer los elementos de interés.

Entorno de desarrollo

Eclipse IDE for Scientific Computing es un entorno de desarrollo que incluye herramientas para C, C ++, Fortran, MPI, OpenMP, OpenACC, un depurador paralelo y aplicaciones de creación, ejecución y supervisión de forma remota, incluye las siguientes herramientas [30]:

- Herramientas de desarrollo C / C ++
- Integración de Git para Eclipse
- Lista de tareas de Mylyn
- Plataforma de herramientas paralelas
- Editores y herramienta XML de Eclipse

El compilador utilizado para compilar los algoritmos es GCC (GNU Compiler Collection), incluye interfaces para C, C ++, Objective-C, Fortran, Ada, Go y otros lenguajes de programación basados en C. Para ejecutar los algoritmos desde la línea de comando en entornos Linux se utilizó archivos Makefile, cuyo contenido está compuesto de órdenes que debe ejecutar la utilidad make para compilar el archivo principal con las dependencias (librerías, archivos de cabecera .h) entre los distintos módulos del proyecto.

Dataset de imágenes

Las imágenes utilizadas en el procesamiento y análisis, tanto con el algoritmo secuencial como con el algoritmo paralelo, son de 24-bits por píxel, y por tanto, con 8 bits por cada canal espectral R, G y B, en formato JPG. Las fotografías se realizaron en un terreno de maíz en la cuarta semana después del sembrío utilizando un dron DJI Mavic 2 Pro (DJI, 2021), capturadas de forma cenital (90 grados) a 10 metros de altura. Las fotografías utilizadas para evaluación de los algoritmos fueron con dimensiones de 1920 x 1080 (imágenes grandes), 1280 x 720 (imágenes medianas) y 640 x 427 (imágenes pequeñas). El dataset de imágenes consta de 30 imágenes propias y se encuentran disponible en https://n9.cl/64ydj.

Equipo de procesamiento

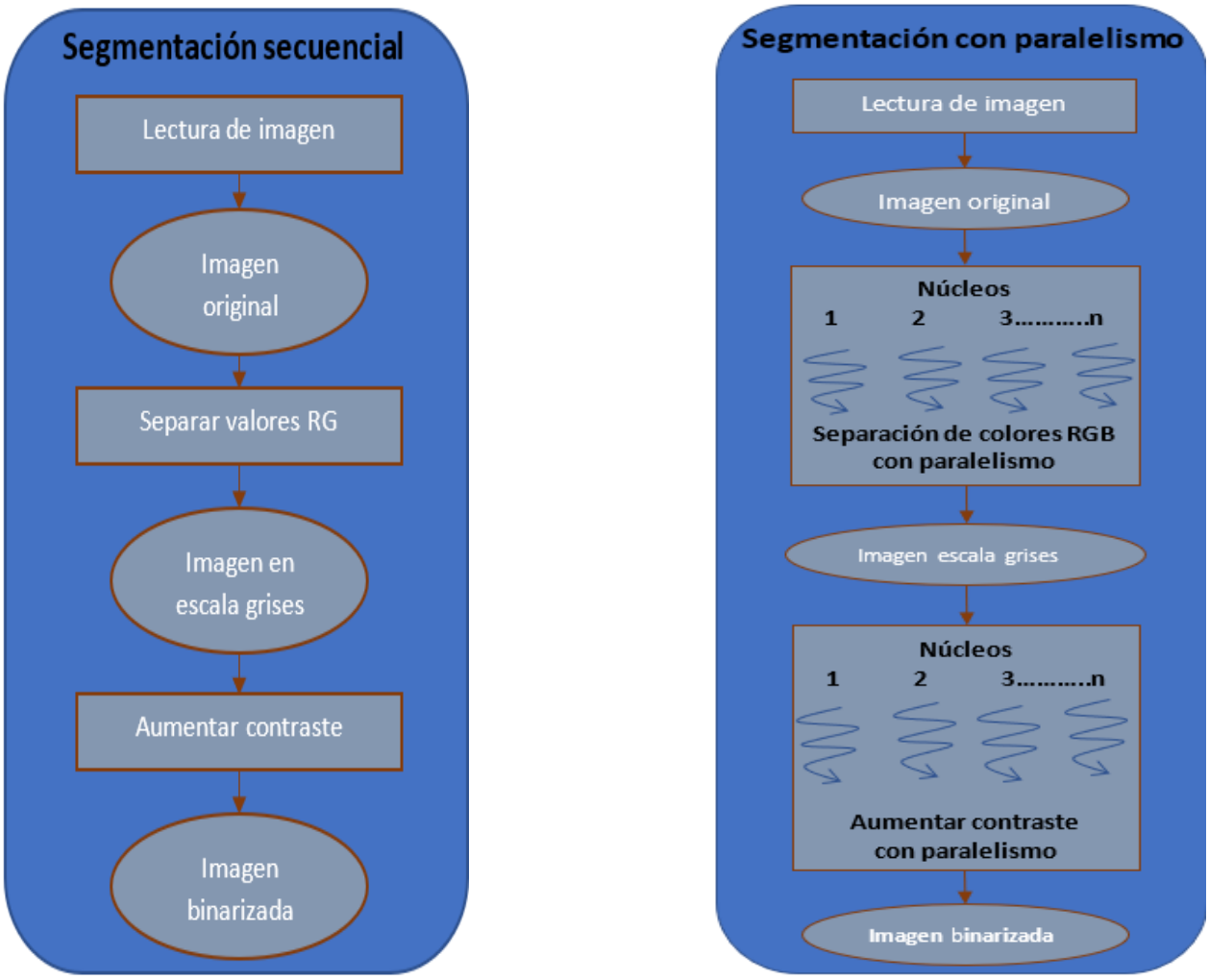
Las pruebas del algoritmo serial y paralelo, se realizaron en una Laptop HP EliteBook 14 Noteboox con Linux Ubuntu 18.04 de 64 bits, procesador Intel Core I7, 16 MB de memoria RAM. Las características técnicas del computador y del CPU utilizado para el procesamiento serial y paralelo se presentan en la Tabla 1:

Tabla 1: Especificaciones técnicas de Laptop con Ubuntu Linux y Procesador multicore-CPU

Detalle	Información
Sistema operativo	Ubuntu 18.04 bionic
Núcleo de Linux	5.4.0-87-generic
Memoria RAM	8 Gb
Arquitectura	X86_64B
Procesador	Intel® Core™ i7-10510U CPU @ 1.80 Ghz
Núcleos	4
Procesadores lógicos	8
Fabricante	Hewlett-Packard

Algoritmo de segmentación

El algoritmo propuesto se implementó con funciones personalizadas para diferenciar la vegetación presente en las imágenes digitales. La Figura 1 presenta el proceso de segmentación del algoritmo secuencial. Para la implementación del algoritmo paralelo se consideró las mismas fases del secuencial y se aplicó paralelismo en las secciones de repetición mediante sentencias de OpenMP.



(a) Algoritmo secuencial (b) Algoritmo con paralelismo

Figura 1: Fases algoritmo secuencial (a) y paralelo (b) para segmentar imágenes digitales

En la lectura de la imagen no es necesario aplicar paralelismo debido a que lee y verifica la ruta. En el Algoritmo 1 se detalla los pasos básicos para cargar la imagen digital desde una ruta específica. La función leer imagen recibe como parámetro un puntero con la ruta de la imagen a ser procesada. Si la imagen existe, se retorna la imagen original y se almacena en un puntero tipo matriz.

Algoritmo 1. Método para leer la imagen offline desde una ruta existente y verificar existencia en la ruta especificada.

```
leer_imagen (* nombreimagen)
Inicio
  Declarar variables para características de la imagen
  Leer imagen y guardar en variable tipo FILE
  Si existe imagen
     Guarda los datos de la imagen
     Asignar espacio de memoria para la imagen
  sino
     Mostrar mensaje "No existe imagen"
  Fin si
  Retornar imagen
Fin
```

En las fases de separación y aumentar contraste se utilizó paralelismo para recorrer los pixeles almacenados en memoria y posteriormente convertir a escala de grises (Algoritmo 2) y aumentar el contraste de los pixeles similares o repetidos (Algoritmo 3). Al final de cada fase se obtiene una nueva imagen con características diferentes dependiendo de las operaciones matemáticas aplicadas.

Algoritmo 2. Método para leer la imagen original y modificar los pixeles RGB a escala de grises mediante una operación matemática.

```
separar_valores(char * imagenoriginal)
Inicio
  Declarar variables para asignar ancho, alto y número de pixeles de
  la imagen
  //sección paralela
  #ifdef PARALLEL
  #pragma omp Parallel for private (r,g,b) num_cores(PARALLEL)
  #endif
  Mientras pixeles<ancho*alto de la imagen
  Inicio
     Leer datos de la imagen desde memoria
     Asignar a cada pixel el respectivo valor RGB
     Convertir los pixeles RGB a gris (200*r + 150*g + 100*b)/1000
     Crear la imagen escala de gris
  Fin mientras
  Retornar imagen gris
fin
```

Algoritmo 3. Método para leer la imagen en escala de grises, agrupar los pixeles similares y modificar el contraste de cada píxel mediante una operación matemática.

```
binarizar_imagen (* imagengris)
Inicio
  Declarar variables para asignar ancho, alto de la imagen
  gris
  Reservar memoria para imagen gris
  //sección paralela
  #ifdef PARALLEL
  #pragma omp Parallel for private (pixel) num_cores(PARALLEL)
  #endif
  Mientras pixeles<ancho*alto imagen gris hacer
  Inicio
     Leer datos de la imagen desde memoria
     Recuperar cada pixel de la imagen gris
     Agrupar pixeles similares o repetidos
  Fin mientras
  //aumentar contraste a la imagen con paralelismo
  #ifdef PARALLEL
  #pragma omp Parallel for private (pixel) num_cores(PARALLEL)
  #endif
  Mientras pixeles<ancho*alto imagen gris hacer
  Inicio
     Leer pixels imagen gris
     Aumentar contraste a cada píxel (250*valorpixel) /
     (ancho * alto)
  Fin mientras
Retornar imagen binaria
Fin
```

Evaluación de algoritmos paralelos

Existen varios criterios que se pueden utilizar con el fin de evaluar la calidad de los algoritmos paralelos. Los criterios usados en el presente trabajo son los recomendados por [31].

El tiempo de ejecución es un buen indicador de la funcionalidad de un algoritmo paralelo. La medición del tiempo de ejecución está representada por el conteo de las unidades de segundos comprendidas entre el momento de inicio y termino de ejecución del algoritmo para la segmentación de imágenes agrícolas.

En computación paralela la rapidez se refiere a que tan rápido es un algoritmo en comparación con otro que resuelve el mismo problema a través de la medición de tiempos. Para determinar la rapidez se utilizó la rapidez absoluta (Sp); la misma que es la razón de los tiempos entre el algoritmo serial (Ts) y el tiempo del algoritmo paralelizado (Tp). La

medición de rapidez está determinada por la ecuación 1.

$$Sp = \frac{Ts}{Tp} \quad (1)$$

Donde Ts es el tiempo de ejecución del algoritmo secuencial y Tp es el tiempo de ejecución del algoritmo paralelo con p procesadores.

En computación se considera eficiencia (Ep) a la proporción del tiempo que se pasa desarrollando trabajo útil. La medición de la eficiencia está determinada por la ecuación 2.

$$Ep = \frac{Sp}{p} \quad (2)$$

Donde Sp es la rapidez previamente calculada del algoritmo y p es el número de procesadores utilizados en la ejecución del algoritmo.

El costo (Cp) de un algoritmo paralelo es el límite sobre un total de operaciones ejecutadas colectivamente por los procesadores. La medición del costo está determinado por la ecuación 3.

$$Cp = \frac{Ts}{Ep} \quad (3)$$

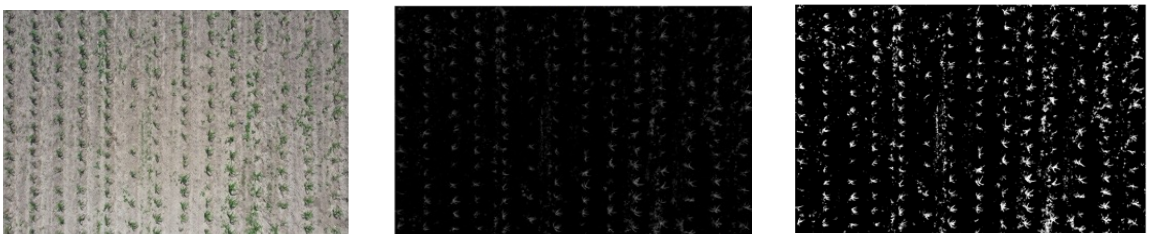
Donde Ts es el tiempo de ejecución del algoritmo serial y Ep es la eficiencia obtenida dependiendo de la cantidad de núcleos.

Resultados y Discusión

La evaluación de los algoritmos implementados se realizó las imágenes del dataset de la sección 2.2. Las imágenes fueron seleccionadas al azar 10 por cada medida (grande, mediana y pequeña) dependiendo del tamaño en pixeles.

Compilación Algoritmos

Para la ejecución del programa se realizó desde la terminal de Linux mediante el comando make. Este comando ejecuta el contenido del archivo Make file para compilar el código fuente y generar los archivos de salida .out. En la Figura 2 se presenta la compilación de los algoritmos serial y paralelo para procesar las imágenes seleccionadas. La ejecución de los algoritmos se realiza siguiendo las fases indicadas en la Figura 1.



(a)Imagen original *(b) Imagen en escala de grises* *(c) Imagen binaria*

Figura 2: Ejecución de las fases algoritmo secuencial y paralelo para segmentar imágenes digitales

Tiempos de ejecución

El tiempo de ejecución es el tiempo que tarda un algoritmo en completar su tarea. En los algoritmos paralelos, se debe tener en cuenta el tiempo de comunicación entre los procesadores. En la Tabla 2 se presentan los tiempos de ejecución promedio de las imágenes de las tres dimensiones evaluadas.

Tabla 2. Tiempos promedio evaluados para las imágenes de las tres dimensiones seleccionadas.

Tamaño imagen	Tiempos de ejecución (segundos)				
	Secuencial	2 núcleos	4 núcleos	6 núcleos	8 núcleos
1920 x 1280	1.59	0.93	0.61	0.57	0.53
1280 x 853	1.13	0.72	0.47	0.32	0.28
640 x 427	0.73	0.42	0.27	0.22	0.19

En la Figura 3 se muestra el comportamiento del algoritmo secuencial en relación con el algoritmo paralelo considerando el número de núcleos de procesador. Los tiempos de ejecución disminuyen dependiendo de número de núcleos utilizados.

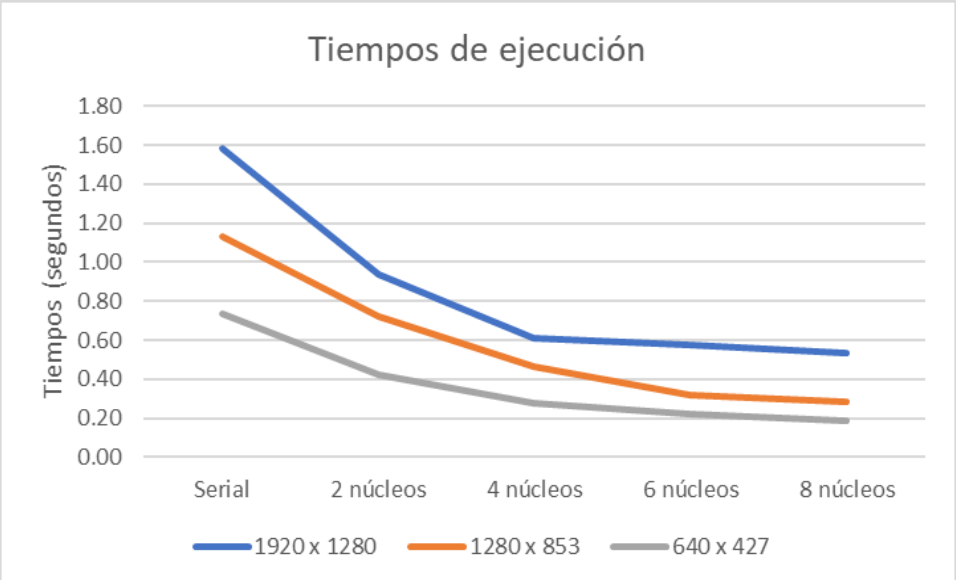


Figura 3. Tiempos de ejecución del algoritmo secuencial y paralelo.

Según los datos presentados en la Tabla 2 y Figura 3, el tiempo de ejecución no es proporcional al incremento de núcleos. La evaluación con 2 núcleos se obtiene una reducción del 41.07%, con imágenes grandes y 42.49% con imágenes pequeñas; pero con mayor cantidad de núcleos el porcentaje es inferior debido a la sobrecarga de los datos en la memoria. Al compartir memoria los algoritmos coordinan y sincronizan los núcleos para garantizar que no se produzcan conflictos, por ende, requiere mayor tiempo en la realización de las tareas asignadas.

Aceleración (Speed-up)

La aceleración se define como la relación entre el tiempo de ejecución de un algoritmo secuencial y el tiempo de ejecución de un algoritmo paralelo. Los valores de la aceleración obtenidos de los algoritmos después de aplicar la ecuación 1 para las imágenes procesadas se presentan en la Tabla 3.

Tabla 3. Aceleración promedio evaluada para las imágenes de las tres dimensiones seleccionadas.

Tamaño imagen	Aceleración (segundos)				
	Serial	2 núcleos	4 núcleos	6 núcleos	8 núcleos
1920 x 1280	1.0000	1.70	2.59	2.76	2.97
1280 x 853	1.0000	1.57	2.43	3.52	4.02
640 x 427	1.0000	1.74	2.67	3.31	3.96

En la Figura 4 se presenta el comportamiento de la aceleración del algoritmo secuencial en relación con el algoritmo paralelo considerando el número de núcleos de procesador. La aceleración incrementa a medida que se utiliza mayor cantidad de núcleos.

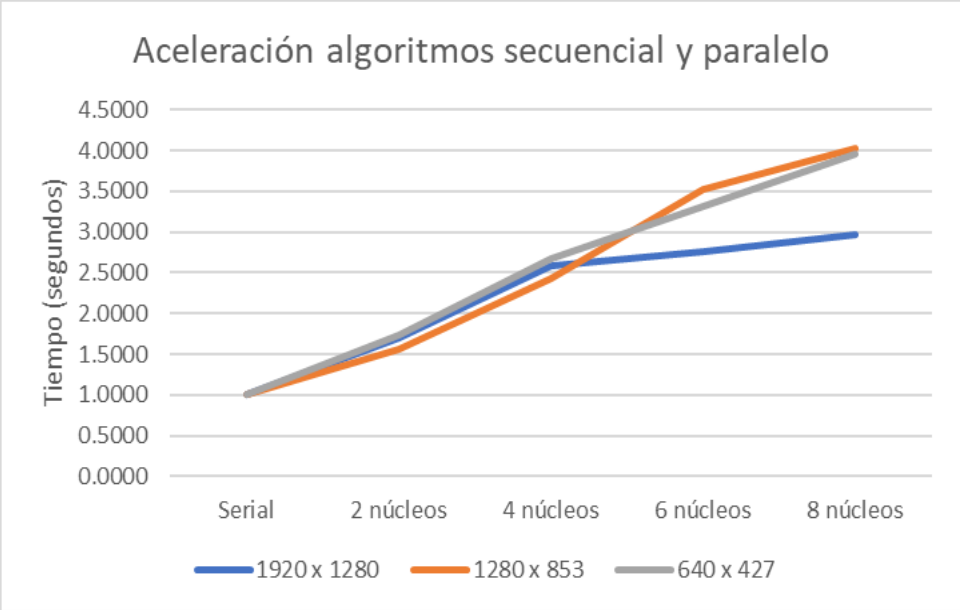


Figura 4. Aceleración del algoritmo paralelo en relación con el algoritmo secuencial

Según los datos presentados en la Tabla 3 y Figura 4, la aceleración máxima obtenida con imágenes medianas es de 4.02 segundos utilizando 8 núcleos del procesador. La aceleración calculada con 8 núcleos se obtiene un incremento de 3 veces más comparada con el algoritmo secuencial. Una aceleración ideal sería igual al número de procesadores utilizado, en este caso el algoritmo implementado no cumple con esta condición.

Eficiencia

La eficiencia se identifica como la porción de tiempo que los elementos de proceso se dedican a trabajo útil, el valor oscila entre 0 y 1. Los valores de eficiencia obtenidos del algoritmo paralelo aplicado la ecuación 2 se presentan en la Tabla 4.

Tabla 4. Eficiencia promedio evaluada para las imágenes de las tres dimensiones seleccionadas.

Tamaño imagen	Eficiencia (segundos)				
	Serial	2 núcleos	4 núcleos	6 núcleos	8 núcleos
1920 x 1280	0.0000	0.85	0.65	0.46	0.37
1280 x 853	0.0000	0.78	0.61	0.59	0.50
640 x 427	0.0000	0.87	0.67	0.55	0.49

En la Figura 5 se presenta el comportamiento de la eficiencia del algoritmo paralelo en relación con el algoritmo secuencial considerando el número de núcleos de procesador.

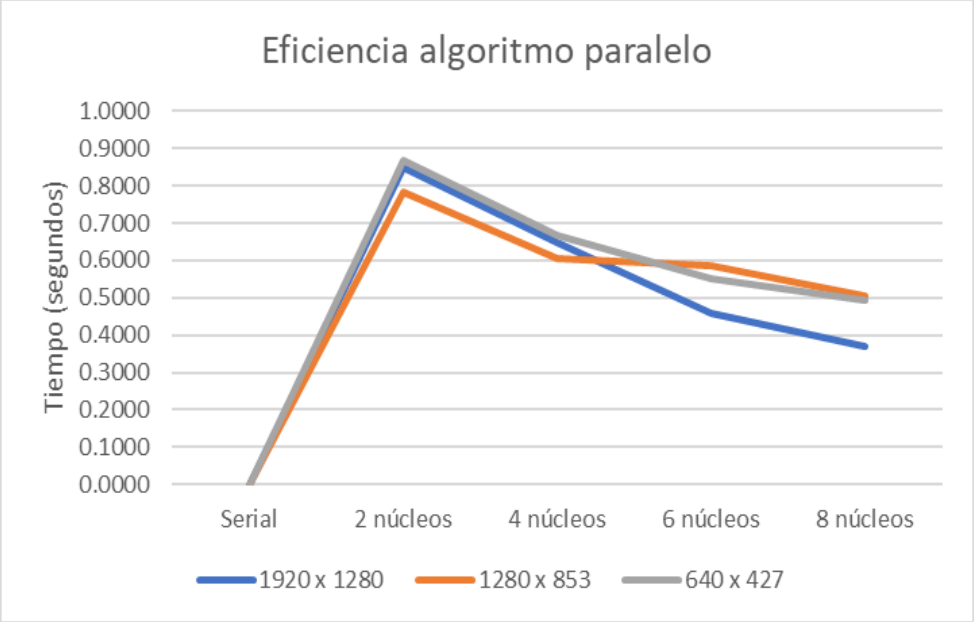


Figura 5. Aceleración del algoritmo paralelo con relación al algoritmo secuencial

Según los datos presentados en la Tabla 4 y Figura 5 se obtiene mejor eficiencia cuando se utiliza 2 núcleos en el procesamiento de imágenes de las 3 dimensiones. Con mayor cantidad de núcleos la eficiencia del algoritmo decrece por la sobrecarga de tareas de procesamiento en cada núcleo. Conceptualmente es posible alcanzar un valor de eficiencia ideal (1) al incrementar número de núcleos; sin embargo, en la práctica la eficiencia máxima en promedio supera el 50% con relación al algoritmo secuencial.

Costo computacional

El costo computacional es la métrica que representa el total de operaciones realizadas durante el procesamiento del algoritmo. Los valores de costo computacional obtenidos del algoritmo paralelo aplicado la ecuación 3 se presentan en la Tabla 5.

Tabla 5. Costo computacional promedio evaluado para las imágenes de las tres dimensiones seleccionadas.

Tamaño imagen	Costo computacional (operaciones)				
	Serial	2 núcleos	4 núcleos	6 núcleos	8 núcleos
1920 x 1280	1.0000	1.87	2.45	3.44	4.26
1280 x 853	1.0000	1.44	1.86	1.93	2.25
640 x 427	1.0000	0.84	1.10	1.33	1.48

En la Figura 6 se presenta el comportamiento del costo computacional del algoritmo paralelo en relación con el algoritmo secuencial considerando el número de núcleos de procesador.

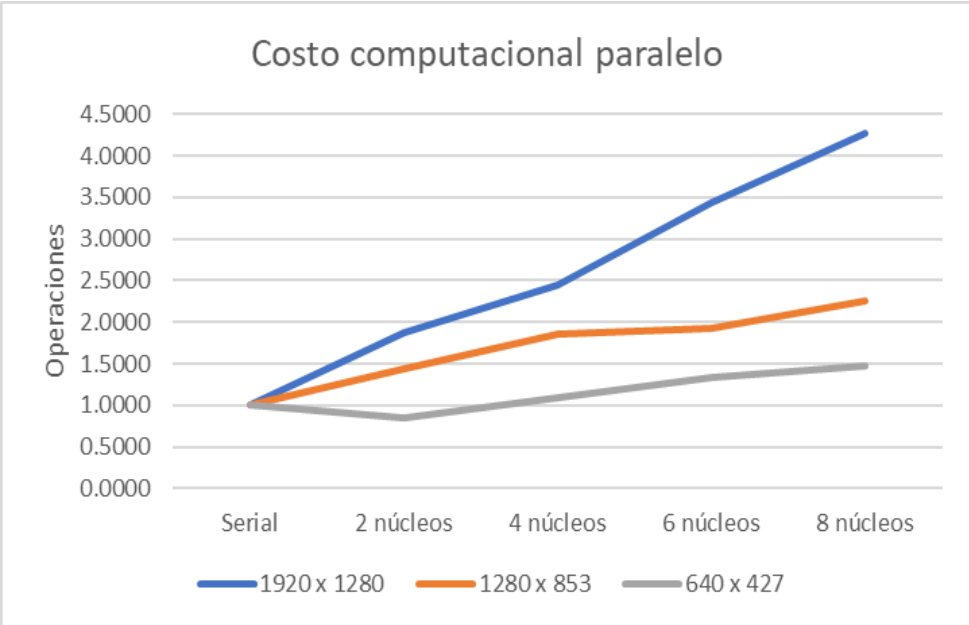


Figura 6. Costo computacional del algoritmo paralelo en relación con el algoritmo secuencial

Según los datos presentados en la Tabla 5 y Figura 6 el mayor costo computacional se obtiene cuando se procesan imágenes grandes, si bien las imágenes grandes tienen mayor cantidad de pixeles, la calidad de estos es inferior con relación a las imágenes pequeñas. En la evaluación se muestra que conforme incrementa el número de núcleos también incrementa el número de operaciones del algoritmo paralelo por la cantidad de pixeles de cada imagen procesada.

Conclusiones y trabajos futuros

Las imágenes obtenidas con drones son insumos importantes en la agricultura de precisión. Los drones son vehículos aéreos no tripulados, equipados con cámaras y sensores que pueden capturar imágenes de alta resolución de campos de cultivo y terrenos agrícolas. Estas imágenes se utilizan para realizar análisis de datos y obtener información valiosa sobre el estado de los cultivos y las condiciones del suelo.

En la agricultura de precisión, la segmentación de imágenes digitales se utiliza para identificar y delimitar las áreas de cultivo, así como para detectar y medir las características de las plantas y los cultivos. La segmentación de imágenes se utiliza en la agricultura de precisión para ayudar a los agricultores a tomar decisiones informadas sobre el uso de recursos y la gestión de los cultivos.

La visión por computador es una técnica de la inteligencia artificial que se utiliza para procesar y analizar imágenes digitales y extraer información útil de ellas. En la agricultura de precisión, la visión por computador se utiliza para analizar imágenes de campos de cultivo y proporcionar información valiosa sobre la salud de las plantas, la calidad del suelo y otros factores que afectan la producción de los cultivos.

OpenMP se define como una API adecuada para el desarrollo de algoritmos paralelos en arquitectura Multicore. El paralelismo utiliza los hilos disponibles y la memoria compartida para distribuir de mejor manera las tareas en cada núcleo del procesador. Las sentencias de programación incluidas en OpenMP permiten ser utilizadas para múltiples códigos sobre diferentes tecnologías disponibles. Los algoritmos que utilizan memoria compartida son útiles en situaciones en las que se necesita procesar grandes conjuntos de datos en paralelo, como en aplicaciones de procesamiento de imágenes digitales. Sin embargo, es importante tener en cuenta que el rendimiento de los algoritmos paralelo puede depender en gran medida de la arquitectura del sistema y de la forma en que se diseñan y programan.

Las métricas definidas como medidas de rendimiento permitieron evaluar el algoritmo paralelo y medir el rendimiento en términos de velocidad y eficiencia. Los resultados obtenidos verifican la mejora del algoritmo paralelo con relación al algoritmo serial porque disminuyen los tiempos de ejecución en un 72 % aproximadamente (Tabla 2 y Figura 3) cuando se ejecuta el algoritmo con 8 núcleos utilizando imágenes con las tres dimensiones seleccionadas. Con respecto a la aceleración, el algoritmo paralelo implementado se obtuvo un promedio de 3.65 veces más rápido (Tabla 3 y Figura 4) comparado con el algoritmo secuencial, lo que permite justificar la implementación en arquitecturas multicore. La eficiencia calculada del algoritmo paralelo alcanza un valor máximo en promedio (Tabla 4 y Figura 5) de 0.83 utilizando 2 núcleos y mínimo 0.46 utilizando 8 núcleos, lo que evidencia la mejora con relación al algoritmo secuencial.

Como trabajos futuros se prevé realizar aplicativos para discriminar las malezas mediante procesamiento paralelo, utilizando arquitecturas heterogéneas y técnicas de inteligencia artificial en lenguajes de programación compatible.

Referencias

[1] J. M. Pajares-Martinsanz, Gonzalo; De La Cruz-García, Visión por Computador Imágenes Digitales y Aplicaciones, Paracuello. Madrid, 2008.

[2] A. Martínez-Rodríguez., "Sistema de procesamiento de imágenes RGB aéreas para agricultura de precisión," Univ. Cent. "Marta Abreu" Las Villas Trab. DIPLOMA, p. 54, 2016.

[3] J. Fang, C. Huang, T. Tang, and Z. Wang, "Parallel programming models for heterogeneous many-cores: a comprehensive survey," CCF Trans. High Perform. Comput., vol. 2, no. 4, pp. 382–400, 2020, doi: 10.1007/s42514-020-00039-4.

[4] G. Barlas, "Chapter 1 - Introduction," in Multicore and GPU Programming, G. Barlas, Ed. Boston: Morgan Kaufmann, 2015, pp. 1–26.

[5] M. Pusdá-Chulde, A. De Giusti, E. Herrera-Granda, and I. García-Santillán, "Parallel CPU-Based Processing for Automatic Crop Row Detection in Corn Fields," in Artificial Intelligence, Computer and Software Engineering Advances, 2021, pp. 239–251.

[6] A. A. Briceño Coronado, Algoritmos paralelos de visión computacional Para reconstrucción tridimensional. México: Editorial Académica Española, 2012.

[7] L. E. Sucar and Gómez Giovani, Visión Computacional, 1st ed. Instituto Nacional de Astrofíisica, Óptica y Electrónica, 2015.

[8] Z. J. Czech, Introduction to Parallel Computing. Cambridge University Press, 2017.

[9] A. Grama, A. Gupta, G. Karypis, V. Kumar, and A. Wesley, Introduction to Parallel Computing, Second Edition. Pearson Education Limited, 2003.

[10] S. Zhang, W. Li, Z. Jing, Y. Yi, and Y. Zhao, "Comparison of Three Different Parallel Computation Methods for a Two-Dimensional Dam-Break Model," Math. Probl. Eng., vol. 2017, p. 1970628, 2017, doi: 10.1155/2017/1970628.

[11] G. Slabaugh, R. Boyes, and X. Yang, "Multicore image processing with openMP," IEEE Signal Process. Mag., vol. 27, no. 2, pp. 134–138, 2010, doi: 10.1109/MSP.2009.935452.

[12] E. García and F. Flego, "Agricultura de Precisión," 2015. Accessed: Dec. 18, 2018. [Online]. Available: https://www.palermo.edu/ingenieria/downloads/pdfwebc&T8/8CyT12.pdf.

[13] Agricultura de Precisión, "¿Qué es la Agricultura de Precisión y cómo iniciarse en ella? - Agricultura de Precision para el Desarrollo," 2022. http://agriculturadeprecisionparaeldesarrollo.com/que-es-la-agricultura-de-precision-y-como-iniciarse-en-ella/ (accessed Sep. 19, 2022).

[14] F. Leiva, "La agricultura de precisión: una producción más sostenible y competitiva con visión futurista," Jan. 2003.

[15] J. Echevarría, "Tarjetas gráficas para acelerar el cómputo complejo," C&T - Univ. Palermo, 2008, Accessed: Mar. 06, 2018. [Online]. Available: http://www.palermo.

edu/ingenieria/downloads/pdfwebc&T8/8CyT08.pdf.

[16] M. Pusdá-Chulde, F. Salazar-Fierro, L. Sandoval-Pillajo, E. Herrera-Granda, I. García-Santillán, and A. De Giusti, Image Analysis Based on Heterogeneous Architectures for Precision Agriculture: A Systematic Literature Review, vol. 1078. 2020.

[17] Z. Tsai, "Processing AI at the Edge: GPU, VPU, FPGA, ASIC Explained - ADLINK Blog," ADLINK, 2021. https://blog.adlinktech.com/2021/02/19/embedded-hardware-processing-ai-edge-gpu-vpu-fpga-asic/ (accessed Sep. 21, 2022).

[18] A. P. Marques Ramos et al., "A random forest ranking approach to predict yield in maize with uav-based vegetation spectral indices," Comput. Electron. Agric., vol. 178, no. September, p. 105791, 2020, doi: 10.1016/j.compag.2020.105791.

[19] V. Partel, S. Charan Kakarla, and Y. Ampatzidis, "Development and evaluation of a low-cost and smart technology for precision weed management utilizing artificial intelligence," Comput. Electron. Agric., vol. 157, no. November 2018, pp. 339–350, 2019, doi: 10.1016/j.compag.2018.12.048.

[20] I. García-Santillán and G. Pajares, "Detección automática de líneas de cultivo: estado del arte y futuras perspectivas," Av. y Apl. Sist. Intel. y nuevas Tecnol., no. 54, pp. 381–398, 2016.

[21] B. Li et al., "Above-ground biomass estimation and yield prediction in potato by using UAV-based RGB and hyperspectral imaging," ISPRS J. Photogramm. Remote Sens., vol. 162, no. December 2019, pp. 161–172, 2020, doi: 10.1016/j.isprsjprs.2020.02.013.

[22] D. Beltrán, "Automatización en la Agricultura: un caso de aplicación artificial , control de calidad en semilleros," 2014, no. September, pp. 1–7, doi: 10.13140/2.1.1299.3922.

[23] D. C. Garcia, G. Á. Martínez, and M. E. Garcia, "Raspberry Pi y Arduino: semilleros en innovación tecnológica para la agricultura de precisión," Informática y Sist. Rev. Tecnol. la Informática y las Comun., vol. 2, no. 1, pp. 74–82, Jan. 2018, doi: 10.33936/ISRTIC.V2I1.1134.

[24] P. Radoglou-Grammatikis, P. Sarigiannidis, T. Lagkas, and I. Moscholios, "A compilation of UAV applications for precision agriculture," Comput. Networks, vol. 172, no. February, p. 107148, 2020, doi: 10.1016/j.comnet.2020.107148.

[25] A. Mukherjee, S. Misra, and N. S. Raghuwanshi, "A survey of unmanned aerial sensing solutions in precision agriculture," J. Netw. Comput. Appl., vol. 148, p. 102461, 2019, doi: 10.1016/j.jnca.2019.102461.

[26] HEMAV, "Teledetección con drones Vs Teledetección satelital," 2022. https://hemav.com/agricultura-de-precision-teledeteccion-satelital-vs-teledeteccion-con-drones/ (accessed Sep. 19, 2022).

[27] Z. Luo, W. Yang, Y. Yuan, R. Gou, and X. Li, "Semantic segmentation of agricultural images: A survey," Inf. Process. Agric., Feb. 2023, doi: 10.1016/J.INPA.2023.02.001.

[28] M. Jaroš et al., "Implementation of K-means segmentation algorithm on Intel Xeon

Phi and GPU: Application in medical imaging," Adv. Eng. Softw., vol. 103, pp. 21–28, 2017, doi: 10.1016/j.advengsoft.2016.05.008.

[29] E. Raei, A. Akbari Asanjan, M. R. Nikoo, M. Sadegh, S. Pourshahabi, and J. F. Adamowski, "A deep learning image segmentation model for agricultural irrigation system classification," Comput. Electron. Agric., vol. 198, p. 106977, Jul. 2022, doi: 10.1016/J.COMPAG.2022.106977.

[30] Eclipse, "Eclipse IDE for Scientific Computing," 2022. https://www.eclipse.org/downloads/packages/release/2022-12/r/eclipse-ide-scientific-computing (accessed Oct. 02, 2022).

[31] A. A. Briceño-Coronado, "Algoritmos paralelos de visión computacional para reconstrucción tridimensional," 26.09.2012, 2012.