Revista Cubana de Ciencias Informáticas
Vol. 16, No. 4, Octubre-Diciembre, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 149-167

Tipo de artículo: Artículo original
Temática: Desarrollo de aplicaciones informáticas
Recibido: 18/08/2022 | Aceptado: 06/09/2022

Algoritmo para el procesamiento paralelo de datos de radar de seguimiento mediante Odroid-N2

Algorithm for parallel processing of tracking radar data using Odroid-N2

Alian Ernesto Matos Rodríguez [1*] https://orcid.org/0000-0001-7333-7411
Leandro Zambrano Méndez [2] https://orcid.org/0000-0002-8243-2813
Roberto Sepúlveda Lima [3] https://orcid.org/0000-0002-9451-6395
Humberto Díaz Pando [2] https://orcid.org/0000-0003-1591-8781

[1] Centro de Investigación y Desarrollo Naval. Calle Estrada Palma No. 13, Casa Blanca, Regla, La Habana, Cuba. aematos@nauta.cu

[2] Universidad Tecnológica de La Habana "José Antonio Echeverría", CUJAE. Calle 114 No. 11901, Marianao, La Habana, Cuba. {lzambrano,hdiazp}@ceis.cujae.edu.cu

[3] Ministerio de Educación Superior de la República de Cuba. Calle 23 No. 565, Vedado, La Habana, Cuba. sepulveda@mes.gob.cu

*Autor para la correspondencia. (aematos@nauta.cu; alianmatos1988@gmail.com)

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 4, Octubre-Diciembre, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 149-167

RESUMEN

Los radares son dispositivos dedicados comúnmente a la detección y rastreo de objetivos. Existen diversos tipos de radares, entre los más comunes se encuentran los de búsqueda y los de seguimiento. Una de las diferencias entre estos es el período de actualización de la información, los primeros lo realizan en el orden de los segundos, mientras que los radares de seguimiento en décimas de milisegundos. En la actualidad, los radares modernos emplean la conversión de la señal analógica en digital, para lo cual se han desarrollado modernizaciones de los radares analógicos utilizando diversas plataformas de hardware. Una parte de los radares de seguimiento en Cuba se sustentan con tecnología analógica, los cuales están sometidos a largos períodos de mantenimiento al presentar inestabilidad durante su funcionamiento. A partir de lo antes mencionado surge la necesidad de desarrollar una solución que permita el procesamiento y representación de los datos en una pantalla digital. La principal contribución de este trabajo es el desarrollo de un algoritmo para procesamiento y representación de los datos de un radar de seguimiento, que cumple con las exigencias de tiempo empleando la placa Odroid-N2. El algoritmo propuesto emplea los principios de programación paralela, mediante la implementación del patrón de programación paralela conocido como segmentación de cauce. Los experimentos realizados arrojaron que la solución propuesta disminuye los tiempos de ejecución en relación a su variante secuencial y satisface el requisito de tiempo de respuesta.

Palabras clave: Radar de seguimiento; programación paralela; patrón paralelo segmentación de cauce; Odroid-N2.

ABSTRACT

Radars are devices commonly dedicated to target detection and tracking. There are various types of radars, including search and tracking radars. One difference between both types is the period information update, in the search radars in order of seconds, while the tracking radars in tenths of milliseconds. Nowadays, modern radars employ in some way the conversion of the analog signal into digital. Modernizations of analog radars have been developed using various hardware platforms. A part of the tracking radars in Cuba are supported by analog technology, which are subjected to long periods of maintenance, as they present

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 4, Octubre-Diciembre, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 149-167

instability during their operation. From the aforementioned, the need arises to develop a solution that allows the processing and representation of data on a digital screen. The main contribution of this work is the development of an algorithm for data processing and representation of tracking radar, which meets the time requirements using the Odroid-N2 board. The proposed algorithm uses scheduling principles, by implementing the parallel scheduling pattern known as pipeline. The experiments carried out showed that the proposed solution decreases the execution times in relation to its sequential variant and satisfies the response time requirement.

Keywords: Tracking radar; parallel programming; pipeline parallel pattern; Odroid-N2.

Introducción

Los radares son dispositivos dedicados comúnmente a la detección y rastreo de objetivos próximos en ambientes terrestres, marítimos y aéreos (William and James, 2014). En su forma básica, un sistema de radar consta de cinco elementos: un transmisor de radio; un receptor de radio sintonizado con la frecuencia del transmisor; dos antenas; y una pantalla, como muestra la Figura 1.

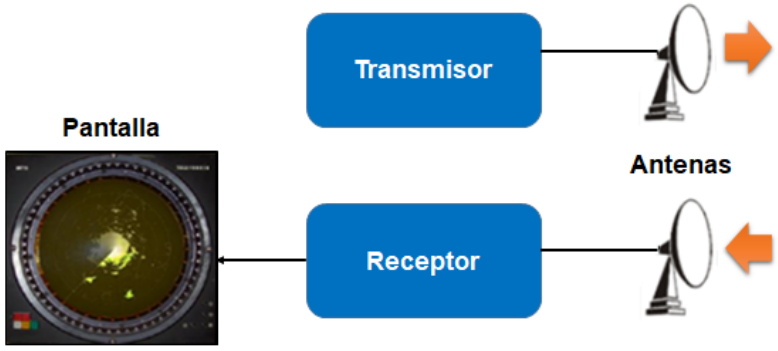

Fig. 1 – Elementos que componen un sistema de radar en su forma básica.

Para detectar la presencia de un objeto (objetivo), el transmisor genera ondas de radio, las cuales son irradiadas por una de las antenas en forma de haz colimado o enfocado. El receptor, mientras tanto, escucha

los ecos de estas ondas de radio, que son captadas por la otra antena. Si se detecta un objetivo, en la pantalla aparece un indicador luminoso que indica su ubicación (George et al., 2014). El PPI es una pantalla en la que los ecos del objetivo se muestran en la posición del plano, con la distancia radial desde el centro que representa el rango y con el ángulo del vector de radio que representa el ángulo de acimut (IEEE, 2017).

Existen diversos tipos de radares, y entre los más comunes se encuentran los de búsqueda y los de seguimiento. Los primeros son utilizados principalmente para la detección inicial de objetivos en un volumen de interés particular, mientras que los segundos tienen como función principal el seguimiento automático de objetivos (IEEE, 2017). Una de las características que marcan las diferencias entre los diferentes tipos de radares, es el período de actualización de la información; por ejemplo, los radares de búsqueda lo realizan en el orden de los segundos, mientras que los radares de seguimiento lo hacen en décimas de milisegundos (George et al., 2014).

En la actualidad los radares modernos emplean la conversión de la señal analógica en digital (George et al., 2014). Para la modernización de los radares analógicos se le añaden otros elementos a los ya observados en la Figura 1, entre los que se encuentran: Procesador Digital de Señales, Procesador de Datos y Visualizador.

El Procesador Digital de Señales se encarga de procesar los ecos de los blancos y de las señales de interferencias recibidas del receptor, con el objetivo de incrementar el nivel de señal y suprimir las interferencias, mientras que el procesador de datos tiene dentro de sus funciones almacenar y procesar los datos con la localización de los blancos detectados para su posterior representación en el visualizador digital.

El procesamiento de los datos y la representación en pantalla digital de la información en los sistemas de radar han sido abordados por disímiles autores en el transcurso de los años, cuyas propuestas se cometan a continuación.

En las propuestas analizadas en (Manasa and Hemalatha, 2015, Kavyashree et al., 2017, Pezhgorski and Lazarova, 2017, Ravindra et al., 2017, Miao et al., 2018, Saputera et al., 2019), no se documenta el algoritmo usado para el procesamiento y representación de los datos. Estas propuestas no presentan un estudio experimental que permita analizar el comportamiento en el uso de los recursos de cómputo en el dispositivo con respecto a los volúmenes de datos procesados. No se especifican las exigencias de tiempo de

actualización de la información de los radares a los cuales va dirigida la solución. Además, se evidencia esencialmente el empleo de la programación secuencial en dichas literaturas.

En las soluciones propuestas en (Manasa and Hemalatha, 2015, Kavyashree et al., 2017, Ravindra et al., 2017, Miao et al., 2018) hicieron uso del lenguaje de programación de alto nivel C++, mientras que en (Pezhgorski and Lazarova, 2017, Saputera et al., 2019) se emplea en lenguaje de nivel medio C, por el rendimiento que ofrecen dichos lenguajes con respecto a otros de alto nivel. Se empleó IDE Qt Creator en (Manasa and Hemalatha, 2015, Kavyashree et al., 2017, Ravindra et al., 2017, Saputera et al., 2019), lo que posibilitó obtener un programa que puede ser ejecutado en diferentes sistemas operativos y plataformas de hardware. Para la visualización de la información del radar en pantalla se aprovecharon las potencialidades que brinda la clase QPainter (QtCompanyLtd., 2021a) y QCustomplot (Eichhammer, 2021) pertenecientes al IDE Qt Creator (Ravindra et al., 2017), mientras que los autores de (Pezhgorski and Lazarova, 2017, Miao et al., 2018) utilizaron para el procesamiento de imágenes la Interfaz de Programación de Aplicaciones (API) OpenGL (Open Graphics Library) (Hearn et al., 2004).

En (Pezhgorski and Lazarova, 2017) utilizaron el principio de programación paralela para garantizar una aceleración basada en la Unidad de Procesamiento Gráfico (GPU, siglas en inglés), lo que permitió la adquisición y conversión en tiempo real de datos de un radar de alta resolución. Los autores en (Manasa and Hemalatha, 2015, Pezhgorski and Lazarova, 2017, Ravindra et al., 2017, Miao et al., 2018, Saputera et al., 2019) usaron como plataforma de hardware computadoras de propósito general, mientras que en (Saputera et al., 2019), además de éstas, se utilizó un Procesador Digital de Señal (DSP, siglas en inglés). Para la comunicación, los autores de (Manasa and Hemalatha, 2015, Kavyashree et al., 2017, Ravindra et al., 2017) emplearon la interfaz Ethernet mediante el Protocolo de Datagrama de Usuario (UDP, siglas en inglés) para el envío y recepción de los datos en tiempo real.

En la actualidad, una parte de los radares de seguimiento en Cuba se sustentan con bloques de procesamiento completamente analógicos y realizan la visualización en monitores de Tubos de Rayos Católicos (TRC). Dichos componentes analógicos son altos consumidores de energía, ocupan gran espacio, generan mucho calor, además de la obsolescencia de las piezas de repuesto e incapacidad de adquirir las mismas en el mercado. Adicionalmente, están sometidos a largos períodos de mantenimiento al presentar inestabilidad durante su funcionamiento.

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 4, Octubre-Diciembre, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 149-167

A partir de la situación planteada, surge la necesidad de desarrollar una solución dirigida a los radares de seguimiento analógicos, que permita el procesamiento y representación de los datos en una pantalla digital, que cumpla con el tiempo máximo de actualización de la información de 40 milisegundos y supere las cualidades del radar analógico. La principal contribución de este trabajo es el desarrollo de un algoritmo para procesamiento y representación de los datos de un radar de seguimiento, que cumple con las exigencias de tiempo empleando la placa Odroid-N2, y supera las cualidades del radar analógico en cuanto a: bajo costo, alto rendimiento, bajo consumo de potencia y fácil implementación.

Métodos o Metodología Computacional

Para desarrollar el algoritmo de procesamiento y visualización de los datos, es necesario tener en cuenta varios elementos tecnológicos, entre los que se encuentran los recursos de hardware tales como: computadoras convencionales, FPGA (Field Programmable Gate Arrays, en lengua inglesa), DSP (Digital Signal Processor, en lengua inglesa), Computadoras de Placa Reducida (SBC, siglas en inglés), entre otras opciones de hardware (Manasa and Hemalatha, 2015, Kavyashree et al., 2017, Pezhgorski and Lazarova, 2017, Ravindra et al., 2017, Miao et al., 2018, Saputera et al., 2019).

Los SBC son computadoras construidas en una sola placa de circuito, con microprocesador(es), memoria, periféricos de entrada / salida (E / S) y otras características requeridas de una computadora funcional. Presentan una amplia gama de características que satisfacen los requisitos planteados por las aplicaciones modernas (Sagkriotis et al., 2019). Estos dispositivos son lo suficientemente potentes para ejecutar sistemas operativos convencionales (Johnston et al., 2018). Su uso ha aumentado de forma vertiginosa en los últimos años debido principalmente a su bajo costo, alto rendimiento, bajo consumo de potencia y fácil implementación (Wazir et al., 2020).

Para la selección de la herramienta de hardware a emplear, se le realizaron pruebas de rendimiento a los SBC con los que se contaba, dígase Odroid-N2 (HardKernel, 2019), Jetson Nano (Nvidia, 2019) y

Raspberry Pi 4 Modelo B (RaspberryPi, 2021). Las pruebas de rendimiento o *benchmarking* fueron de gran importancia para comparar los dispositivos, lo que permitió observar la respuesta de estos ante distintas cargas de trabajo. Atendiendo a que el procesamiento y representación de los datos de un radar son operaciones computacionalmente muy exigentes, las pruebas se enfocaron principalmente en la unidad central de procesamiento mediante la herramienta *Sysbench* (Kopytov, 2012).

Las pruebas de rendimiento a la unidad central de procesamiento de las computadoras de placa reducida, consistió en realizar los chequeos de primalidad para 14000 números primos con uno y cuatro subprocesos de trabajo, respectivamente.

Del análisis de los resultados de las pruebas del CPU se llegó a la conclusión de que tanto el Jetson Nano como el Odroid-N2 tuvieron un mejor rendimiento con respecto a la Raspberry Pi 4. Dado que el Jetson Nano y el Odroid-N2 no se distinguen notablemente en sus desempeños en tiempo de ejecución, se seleccionó este último que es el de menor precio, teniendo en cuenta un escenario en que fuera necesario fabricar varios prototipos.

Diseño de la solución

En la Figura 2 se muestra un esquema del diseño estructural de la solución propuesta, donde se muestra enmarcado en color naranja el bloque funcional desarrollado, conformado por el Odroid-N2 para el procesamiento de los datos y su posterior representación en la pantalla digital PPI, cumpliendo con las exigencias de tiempo de 40 milisegundos impuestas por el radar en cuestión.

El algoritmo secuencial propuesto para el procesamiento en el Odroid-N2 y la visualización de los datos en la pantalla digital, consta de un proceso dividido en tres etapas: recepción de los datos, procesamiento y representación en la pantalla PPI.

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 4, Octubre-Diciembre, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 149-167

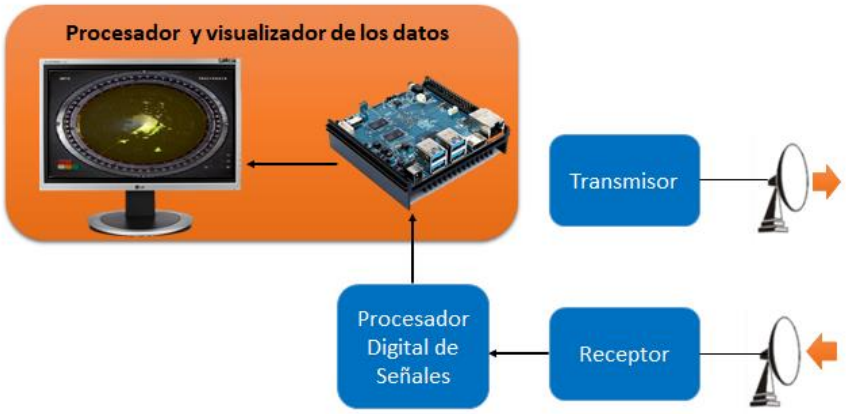

Fig. 2 – Esquema del diseño de la solución propuesta con bloques funcionales digitales.

Implementación de la solución

Para la implementación de la solución se instaló el Sistema Operativo Ubuntu 18.04 en el Odrioid-N2. El software fue desarrollado en el IDE Qt Creator y en el lenguaje de programación C++. De dicho IDE se utilizaron algunas clases propias, entre las que se encuentran: QPainter para el dibujo en Interfaces Gráficas de Usuario (GUI), QUDPSocket para recibir los datagramas mediante el protocolo UDP del módulo de procesamiento digital y la clase QThread para la gestión de los hilos dentro del programa. En el caso del lenguaje C++ se tuvo en cuenta para su empleo su alto rendimiento y compatibilidad con la API OpenMP (Open MultiProcessing, en lengua inglesa), lo que permitió añadir concurrencia al algoritmo, mediante la programación multiproceso de la memoria compartida en la plataforma de hardware seleccionada.

Recepción de los datos

La recepción de los datos provenientes del bloque funcional Procesador Digital de Señales se realizó a través del adaptador gigabit Ethernet del Odroid-N2 empleando el protocolo de comunicación UDP. Para esto se utilizó la clase *QUdpSocket* del IDE Qt Creator que permite enviar y recibir datagramas UDP (QtCompanyLtd., 2021b). Se reciben de dicho bloque funcional digital 13320 bytes cada 500 microsegundos (µs) durante 36 milisegundos, siendo este último el tiempo de duración de la exploración de la antena del sistema de radar en cuestión.

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 4, Octubre-Diciembre, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 149-167

Procesamiento de los datos

Para el procesamiento de los datos recibidos se tuvo en cuenta la resolución de pantalla a utilizar en la representación en el indicador, siendo en esta solución de 1920x1080 px que es la máxima resolución del monitor empleado. A partir de lo antes planteado, se realizó la selección del radio de la pantalla PPI tomando como diámetro el valor 1080 de los pixeles verticales, arrojando como resultado que el radio para la resolución propuesta puede tener un valor máximo de 540 px. Teniendo en consideración lo mencionado anteriormente y atendiendo a que además de la circunferencia de la pantalla PPI se visualizarán otras informaciones del radar, se seleccionó un radio de 444 px.

Como la cantidad de muestras a procesar cada 500 ms es mayor que el radio de la pantalla, se tuvieron en cuenta varios métodos para realizar la integración de las muestras y determinar el valor de cada pixel. En este caso como la cantidad de muestras es de 13320 y el radio es de 444 px, el valor de intensidad de cada pixel es el resultado de la integración de 30 muestras. Para dicha integración se analizaron tres métodos: promedio, valor máximo y valor máximo de promedios (Trujillo et al., 2020, Guevara Trujillo et al., 2021). El primero consiste en tomar el valor máximo de las 30 muestras; en el segundo el promedio de todas las muestras y en el tercero se dividen las 30 muestras que representan un píxel, en seis subgrupos de 5 muestras promediados de manera independiente. Luego se escoge el mayor de esos seis valores de promedios, el cual sería el valor de intensidad del píxel. Para asignar el valor 5 a la cantidad de muestras por subgrupo en el método máximo de promedios, se tuvo en cuenta la cantidad de muestras por pulso (n) que compone un objetivo, sea τ_p el ancho del pulso del radar de 200 nanosegundos, f_s la frecuencia de muestreo de 25 MHz y T_s el período de muestreo. Se define T_s según (1) y n según (2), obteniéndose como resultados: $T_s = 40\ ns$ y $n = 5$.

$$T_s = \frac{1}{f_s} \qquad (1)$$

$$n = \frac{\tau_p}{T_s} \qquad (2)$$

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 4, Octubre-Diciembre, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 149-167

Para la selección del método empleado en la integración de las muestras por pixel se realizaron simulaciones en la herramienta LabView (Bitter et al., 2017), con el objetivo de observar el comportamiento del nivel de ruido y del blanco en cada uno de los métodos. En los resultados de la simulación, se observó que el método de promedio tiene como ventaja que disminuye la influencia de los picos de ruido, pero tiene como desventaja que disminuye el nivel del blanco. El método valor máximo tiene como aspecto positivo que mejora el nivel del blanco, pero pueden existir picos de ruido que pueden confundirse con un blanco. El método valor máximo de promedio, a pesar de que los valores no llegan a los extremos de los métodos anteriores, tiene como ventaja que disminuye la influencia de los picos de ruido y por ende se observa mejor el blanco. Atendiendo a lo antes mencionado se seleccionó este último para la integración de las muestras por pixel, generando como resultado 444 bytes que son representados en los 444 px de radio de la pantalla PPI que equivalen a una distancia de 40 km.

Representación en pantalla PPI

Para la representación gráfica en la pantalla PPI se empleó la clase *QPainter* del IDE Qt Creator proporcionando funciones altamente optimizadas para realizar programas con GUI de dibujo (QtCompanyLtd., 2021a). Como se cuenta con el acimut (θ) y la distancia (ρ) de las muestras en la dirección donde se encuentra explorando la antena del radar, para la representación se realiza la conversión tradicional de coordenadas polares a cartesianas según 3 y 4.

$$x = \rho \operatorname{sen}(\theta) \quad (3)$$

$$y = \rho \cos(\theta) \quad (4)$$

Los pares ordenados (x,y) representan coordenadas, que en conjunto con las intensidades que se generaron en el procesamiento de los datos son introducidas como parámetros en las funciones de graficado de la clase QPainter.

Con la culminación de la representación de la información en pantalla se le realizaron pruebas de rendimiento al software para verificar su correcto funcionamiento y así observar los tiempos de respuesta en

la placa Odroid-N2. Los resultados de las pruebas antes mencionadas arrojaron que al ejecutar de forma secuencial las etapas de recepción, procesamiento y visualización de los datos, la aplicación tuvo una demora en tiempo de 54 milisegundos, por lo que no cumplió con la exigencia de tiempo de actualización de la información de 40 milisegundos.

Mejora del rendimiento del algoritmo

Al no cumplir el software secuencial con la exigencia de tiempo planteada, se procedió a identificar los puntos del programa de mayor consumo de tiempo mediante la herramienta de creación de perfiles Gprof (Fenlason and Stallman, 1988, Suau et al., 2021). Del perfil estadístico generado por la herramienta Gprof se concluyó que las funciones que tienen un mayor consumo de tiempo en el programa son: la que se encarga de la recepción de datos, que representó un 52% del tiempo total, seguido de la función de procesamiento de los datos con un 18% y la de representación en la pantalla PPI un 12%.

Paralelización del algoritmo secuencial

Todas las computadoras modernas admiten el paralelismo en el hardware a través de al menos una función paralela, incluidas instrucciones vectoriales, núcleos multiproceso, núcleos de múltiples procesadores, procesadores múltiples, motores gráficos y coprocesadores paralelos (McCool et al., 2012). Una manera de diseñar e implementar de manera eficiente los algoritmos paralelos es a través del uso de patrones de computación paralela y distribuida (McCool et al., 2012). Dichos patrones son construcciones abstractas no sujetos a arquitecturas, lenguajes de programación o sistemas (McCool et al., 2012).

De los patrones estudiados se seleccionó el patrón Segmentación de Cauce. Este consiste en una cadena de procesos conectados de forma tal que la salida de cada elemento en la cadena es la entrada del próximo. Los datos son adquiridos de forma segmentada. Estos pasan por todas las etapas desde la primera hasta la última. Cada etapa realiza una transformación de los datos. Más de una etapa puede estar activa al mismo tiempo, por lo que puede ocurrir paralelismo de tareas (McCool et al., 2012).

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 4, Octubre-Diciembre, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 149-167

Para la implementación del patrón Segmentación de Cauce se utilizó la API OpenMP (*Open MultiProcessing*). Esta API ha sido ampliamente adoptada por la comunidad científica informática (Castillo Reyes et al., 2016). Como se observó con anterioridad, el algoritmo secuencial cuenta con tres etapas, de las cuales se conoce que la recepción es la que tiene un mayor consumo de tiempo en la ejecución, seguido del procesamiento y la representación de los datos, con un 52%, 18% y 12%, respectivamente. Teniendo en cuenta lo antes expuesto se paralelizó el código mediante el patrón Segmentación de Cauce, dividiendo el algoritmo en dos etapas, dígase recepción en la primera, mientras que procesamiento y representación en la segunda etapa. En la Figura 3 se muestra un diagrama de actividades que representa el flujo de ejecución del algoritmo utilizando este patrón.

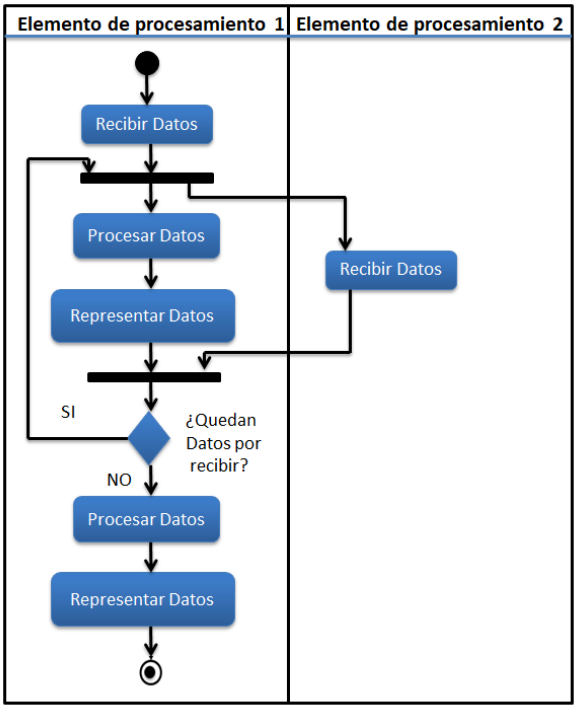

Fig. 3 – Diagrama de actividades que representa el flujo de ejecución del algoritmo paralelo utilizando el patrón segmentación de cauce.

La paralelización del algoritmo secuencial consistió primeramente en lograr la recepción de los datos en la primera etapa del patrón Segmentación de Cauce, siendo esta la de mayor consumo de tiempo, y en paralelo

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 4, Octubre-Diciembre, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 149-167

que se ejecute la segunda etapa formada por el procesamiento y la representación de los datos en la pantalla PPI. Se decidió implementar dicho patrón en sólo dos etapas, atendiendo a que la velocidad de este se verá limitada por el tiempo que demore la ejecución de la etapa más lenta del algoritmo, siendo la de recepción la que demora y condiciona al resto, por lo que, al ejecutar el procesamiento y la representación secuencial en la segunda etapa, que suman un 30% del consumo total del tiempo, no superan el tiempo de la etapa de recepción que es de un 52%.

Al aplicar este patrón, en la primera iteración solo se realizará la recepción de los datos debido a que no se puede ejecutar al algoritmo en paralelo porque todavía no existen datos para el procesamiento y la representación. En las próximas iteraciones sí se ejecutan ambas etapas en paralelo hasta llegar a la última, donde sólo se ejecutará el procesamiento y la representación de los datos ya que se han procesado y representado todos los datos recibidos. El paralelismo que se logra cuando se pone en práctica este patrón se muestra en la Figura 4 donde través del análisis de la figura se podría manejar la hipótesis que el algoritmo paralelo permitirá disminuir los tiempos de ejecución con respecto a su variante secuencial.

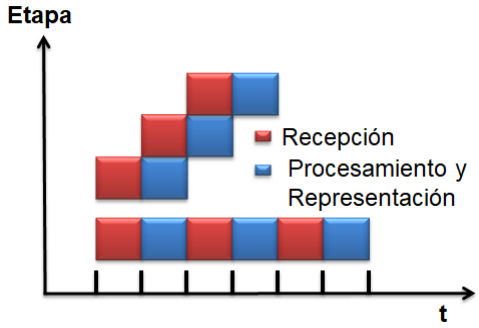

Fig. 4 – Representación de la ejecución paralela del algoritmo mediante el patrón segmentación de cauce con respecto a su variante secuencial.

Resultados y discusión

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 4, Octubre-Diciembre, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 149-167

Se realizó un experimento bajo el principio de repetición para minimizar el error experimental, donde se hicieron 10 ejecuciones a la versión secuencial y paralela, con el objetivo de comprobar si se lograron disminuir los tiempos de ejecución del mismo. Se ejecutaron bajo un escenario de experimentación con las mismas condiciones, dígase SBC Odroid-N2, con Sistema Operativo Ubuntu 18.4, memoria RAM DDR4 de 2 GB, memoria de almacenamiento eMMC de 32 GB e igual señal patrón a procesar y representar en pantalla. Como resultado del experimento se obtuvo que el valor promedio para las 10 ejecuciones de la versión secuencial y paralela fue de 54.97 y 24.3 respectivamente.

El desempeño mostrado por un programa puede tener distintas medidas de evaluación. La medida más común y que resulta más importante dependiendo del fin que tenga el programa, es el tiempo de ejecución, cuanto menos tiempo requiera para ejecutarse, mejor será el rendimiento. Además del tiempo de ejecución como medida de rendimiento, también existen otras métricas como la aceleración (*Speed Up*) y la eficiencia, las cuales se definen a continuación.

La aceleración se define como la relación que existe entre el tiempo de ejecución de un algoritmo empleando un solo procesador y su tiempo de ejecución empleando varios procesadores. Es una métrica que determina cuán rápido es un programa paralelo en comparación con su variante secuencial. Matemáticamente se define como se muestra en la ecuación (5):

$$S_{(P)} = \frac{T_{(1)}}{T_{(P)}} \qquad (5)$$

$$S_{(P)} = \frac{54.97}{24.3} = 2.262$$

Donde:

$S_{(P)}$: Aceleración utilizando P procesadores.

$T_{(1)}$: Tiempo requerido por el sistema con un procesador para resolver el problema en cuestión.

$T_{(P)}$: Tiempo requerido por el sistema con P procesadores para resolver el problema en cuestión.

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 4, Octubre-Diciembre, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 149-167

La eficiencia es definida como la fracción del tiempo que el procesador consume haciendo trabajo útil. Esta muestra cuán bien se ha utilizado los procesadores en la solución de un problema. La fórmula para calcularla se define en (6):

$$E_{(P)} = \frac{S_{(P)}}{P} \qquad (6)$$

$$E_{(P)} = \frac{2.262}{4} = 0.5655$$

Donde:

$E_{(P)}$: Eficiencia.

$S_{(P)}$: Aceleración utilizando P procesadores.

La aceleración obtenida en los experimentos significa que la variante paralela es 2.262 veces más rápida que la variante secuencial, obteniéndose una mejora de aproximadamente un 55.79 %. La eficiencia obtenida es de 0.5655 lo que implica que se aprovecha un 56.55 % los recursos del procesador.

Conclusiones

El algoritmo paralelo propuesto en este trabajo garantiza el procesamiento y la representación digital de la información de un radar de seguimiento en el tiempo establecido.

El diseño de la solución teniendo en cuenta los principios de programación paralela con la API OpenMP, propició una mejora del rendimiento de la solución y un mejor aprovechamiento de los recursos de cómputo de los dispositivos de hardware.

Dicho diseño paralelo en conjunto con las computadoras de placa reducida, contribuyó en la obtención de una solución de alto rendimiento, bajo consumo de potencia y fácil implementación. Además, la arquitectura hardware-software propuesta requiere un bajo costo de implementación, debido al empleo de computadoras de placa reducida y software libre.

Referencias

Bitter, R., Mohiuddin, T. & Nawrocki, M. 2017. *Labview™ Advanced Programming Techniques: Advanced Programming Techniques*, Crc Press.

Castillo Reyes, O., Puente, J. D. L., Modesto, D., Puzyrev, V. & Cela, J. M. 2016. A Parallel Tool For Numerical Approximation Of 3d Electromagnetic Surveys In Geophysics. *Computación Y Sistemas,* 20**,** 29-39.

Eichhammer, E. 2021. *Qcustomplot* [Online]. Available: Https://Www.Qcustomplot.Com/ 2021].

Fenlason, J. & Stallman, R. 1988. Gnu Gprof. *Gnu Binutils. Available Online: Http://Www.Gnu.Org/Software/Binutils*.

George, S. W., Hugh, G. D., Chris, B. J. & Dave, A. 2014. *Stimson's Introduction To Airborne Radar–Third Edition* Scitech

Guevara Trujillo, L., Socarras Hernández, B. N., Zambrano Méndez, L., Hojas-Mazo, W. & Ampuero, M. A. 2021. Sistema Visualizador Para Un Radar De Seguimiento Con Reducción De Costo, Consumo Eléctrico Y Tamaño. *Revista Cubana De Ciencias Informáticas,* 15**,** 76-104.

Hardkernel. 2019. *Odroid-N2* [Online]. Available: Https://Www.Hardkernel.Com/Blog-2/Odroid-N2/ 2021].

Hearn, D., Baker, M. P. & Baker, M. P. 2004. *Computer Graphics With Opengl*, Pearson Prentice Hall Upper Saddle River, Nj:.

Ieee, R. S. P. O. T. 2017. Ieee Standard For Radar Definitions. 1-54.

Johnston, S. J., Basford, P. J., Perkins, C. S., Herry, H., Tso, F. P., Pezaros, D., Mullins, R. D., Yoneki, E., Cox, S. J. & Singer, J. 2018. Commodity Single Board Computer Clusters And Their Applications. *Future Generation Computer Systems,* 89**,** 201-212.

Kavyashree, V., Chaitra, P. M., Prasad, A. & Vinutha, H. 2017. A Radar Target Generator For Airborne Targets. *International Journal Of Science Technology & Engineering,* 3.

Kopytov, A. 2012. Sysbench Manual. *Mysql Ab***,** 2-3.

Manasa, M. & Hemalatha, M. 2015. Desing Of Generic Radar Data Visualizer Using Model View Controller Pattern. *International Journal For Technological Research In Engineering,* 2.

Mccool, M., Reinders, J. & Robison, A. 2012. *Structured Parallel Programming: Patterns For Efficient Computation*, Elsevier.

Miao, Z., Xu, W. & Chen, H. Technology Of Radar Terminal On Researching Software Display. Proceedings Of The 2nd International Conference On Vision, Image And Signal Processing, 2018. 1-5.

Nvidia. 2019. *Jetson Nano Developer Kit* [Online]. Available: Www.Nvidia.Com/.

Pezhgorski, V. & Lazarova, M. Real Time Gpu Accelerated Radar Scan Conversion And Visualization. Proceedings Of The 18th International Conference On Computer Systems And Technologies, 2017. 249-256.

Qtcompanyltd. 2021a. *Qpainter Class* [Online]. Available: Https://Doc.Qt.Io/Qt-5/Qpainter.Html#Details.

Qtcompanyltd. 2021b. *Qudpsocket Class* [Online]. Available: Https://Doc.Qt.Io/Qt-5/Qudpsocket.Html#Details.

Raspberrypi. 2021. *Raspberry Pi 4 Computer Model B* [Online]. Available: Www.Raspberrypi.Org 2022].

Ravindra, C., Rajkumar, S. & Babu, M. S. Design And Implementation Of Radar Console Displays For Multi Object Tracking Radar Using Qt-Ide. 11th International Radar Symposium India, 2017.

Sagkriotis, S., Anagnostopoulos, C. & Pezaros, D. P. Energy Usage Profiling For Virtualized Single Board Computer Clusters. 2019 Ieee Symposium On Computers And Communications (Iscc), 2019. Ieee, 1-6.

Saputera, Y. P., Wahab, M. & Maulana, Y. Y. 2019. Design Of Radar Display Of Indonesian Airspace Monitoring Application. *Telkomnika,* 17**,** 1176-1184.

Suau, A., Staffelbach, G. & Todri-Sanial, A. 2021. Qprof: A Gprof-Inspired Quantum Profiler. *Acm Transactions On Quantum Computing*.

Trujillo, L. G., Rodríguez, A. M. & Méndez, L. Z. 2020. Representación En Tiempo Real De Señales De Radar Empleando Odroid Xu4. *Elektron*, 4**,** 87-92.

Wazir, S., Imran, H. A., Latif, U., Mujahid, U. & Bilal, M. 2020. Single Board Computers (Sbc): The Future Of Next Generation Pedagogies In Pakistan. *Arxiv Preprint Arxiv:2008.06576*.

William, M. L. & James, S. A. 2014. Principles Of Modern Radar, Vol. Iii: Radar Applications. Scitech Publishing, New Jersey.

Conflicto de interés

Contribuciones de los autores

1. Conceptualización: Alian Ernesto Matos Rodríguez
2. Curación de datos: Leandro Zambrano Méndez y Roberto Sepúlveda Lima
3. Análisis formal: Alian Ernesto Matos Rodríguez
4. Adquisición de fondos: -
5. Investigación: Alian Ernesto Matos Rodríguez
6. Metodología: Leandro Zambrano Méndez y Humberto Díaz Pando
7. Administración del proyecto: Alian Ernesto Matos Rodríguez
8. Recursos: Leandro Zambrano Méndez y Roberto Sepúlveda Lima
9. Software: Alian Ernesto Matos Rodríguez
10. Supervisión: Leandro Zambrano Méndez
11. Validación: Leandro Zambrano Méndez y Humberto Díaz Pando
12. Visualización: Alian Ernesto Matos Rodríguez
13. Redacción – borrador original: Alian Ernesto Matos Rodríguez
14. Redacción – revisión y edición: Leandro Zambrano Méndez y Roberto Sepúlveda Lima